THE DEVELOPMENT OF BIO-INSPIRED CORTICAL FEATURE MAPS FOR ROBOT SENSORIMOTOR CONTROLLERS

SAMANTHA V. ADAMS

School of Computing and Mathematics

Faculty of Science and Technology

February 2012

COPYRIGHT NOTICE

THE DEVELOPMENT OF BIO-INSPIRED CORTICAL FEATURE MAPS FOR ROBOT SENSORIMOTOR CONTROLLERS

By

SAMANTHA VANESSA ADAMS

A thesis submitted to the University of Plymouth

in partial fulfilment for the degree of

DOCTOR OF PHILOSOPHY

School of Computing and Mathematics

Faculty of Science and Technology

February 2012

SAMANTHA VANESSA ADAMS

THE DEVELOPMENT OF BIO-INSPIRED CORTICAL FEATURE MAPS FOR ROBOT SENSORIMOTOR CONTROLLERS

ABSTRACT

This project applies principles from the field of Computational Neuroscience to Robotics research, in particular to develop systems inspired by how nature manages to solve sensorimotor coordination tasks. The overall aim has been to build a self-organising sensorimotor system using biologically inspired techniques based upon human cortical development which can in the future be implemented in neuromorphic hardware. This can then deliver the benefits of low power consumption and real time operation but with flexible learning onboard autonomous robots. A core principle is the Self-Organising Feature Map which is based upon the theory of how 2D maps develop in real cortex to represent complex information from the environment. A framework for developing feature maps for both motor and visual directional selectivity representing eight different directions of motion is described as well as how they can be coupled together to make a basic visuomotor system. In contrast to many previous works which use artificially generated visual inputs (for example, image sequences of oriented moving bars or mathematically generated Gaussian bars) a novel feature of the current work is that the visual input is generated by a DVS 128 silicon retina camera which is a neuromorphic device and produces spike events in a frame-free way. One of the main contributions of this work has been to develop a method of autonomous regulation of the map development process which adapts the learning dependent upon input activity. The main results show that distinct directionally selective maps for both the motor and visual modalities are produced under a range of experimental scenarios. The adaptive learning process successfully controls the rate of learning in both motor and visual map development and is used to indicate when sufficient patterns have been presented, thus avoiding the need to define in advance the quantity and range of training data. The coupling training experiments show that the visual input learns to modulate the original motor map response, creating a new visual-motor topological map.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I'd like to dedicate this thesis to my mother, Pamela Hayne who sadly passed away during the final year of my PhD and so did not get to see me complete. She was always very supportive of my lifelong obsession with study and particularly proud when I succeeded in getting a PhD studentship. It is in no small part that my success to date is due to her influence in my early years.

I'd also like to thank my supervisory team, Dr Phil Culverhouse, Dr Thomas Wennekers and Dr Sue Denham for their guidance and allowing me the freedom to develop my own ideas within the original project remit.

Very special thanks are also due to Martin Beck, my Masters supervisor who set me on the path to research and gave me the confidence to apply for a PhD.

Finally I'd like to thank my husband Cris Adams and my dear friends Jon Williams and Penny Marno who have on many occasions provided support and a friendly ear in times of frustration. Also thank you to all my friends and colleagues at the Centre for Robotics and Neural Systems for making such a friendly and fun place to study.

AUTHOR'S DECLARATION

At no time during the registration for the degree of Doctor of Philosophy has the author been registered for any other University award without prior agreement of the Graduate Committee.
This study was financed with the aid of a studentship from the Engineering and Physical Sciences Research Council and a scholarship from the University of Plymouth Graduate School.
During the course of this study I attended the Telluride Neuromorphic Cognition Workshop (2011) as well as several postgraduate courses for generic research skills. Poster and Paper presentations were given at relevant conferences and several papers have been prepared for publication.

Publications:

Adams, S.V., Culverhouse, P.F, Wennekers, T., Bugmann, G. and Denham, S. (2010) 'A Sensori-Motor Model of Arachnid Prey Localisation' in: *Proceedings of the 11th Towards Autonomous Robotic Systems Conference (TAROS 2010)*, Plymouth, UK: 1-6. ISBN: 978-1-84102-263-5

Adams, S.V., Wennekers, T., Bugmann, G., Denham, S. and Culverhouse, P.F (2011) 'Application of Arachnid Prey Localisation Theory for a Robot Sensorimotor Controller', *Neurocomputing, 74(17): 3335-3342.*

Presentation and Conferences Attended:

Towards Autonomous Robotic Systems Conference (TAROS 2010) (paper presentation)

Postgraduate Conference for Computing: Applications and Theory (PCCAT 2011) (poster presentation)

Word Count of main body of thesis: 40390

Signed …….

Date ……

1 Introduction

1.1 Research Topic

An important goal in robotics is to create autonomous machines that can perform basic tasks related to their own maintenance and welfare without human intervention. In short, it would be desirable for them to have more sophisticated human-like capabilities. This raises some major challenges as traditional computing and engineering approaches can only achieve so much. They can be used to mimic human capabilities to some extent but it is difficult to make systems that work in the same way as natural ones do. As we do not fully understand all the neural processing which generates our own behaviour it is often difficult to translate the concepts into traditional approaches. Since the 1990s Artificial Intelligence researchers have advocated a 'situated and embodied' approach to robotics: they propose that more appropriate behaviours can be generated by the robot existing in and interacting with an environment (Brooks, 1990). This ties in with approaches inspired directly by human development where robots acquire capability gradually by learning in an environment (Developmental or Epigenetic Robotics). However, the sensory pre-processing and higher level cognitive processing that is required to achieve such human-like learning capabilities requires significant computing power which is in conflict with the limited energy resources available on an autonomous robot. It should be noted, however, that natural neural systems manage to achieve speed, fault tolerance and flexibility despite having very low power requirements! Therefore, it seems logical to explore in more depth bio-inspired approaches to robotics. In particular, where artificial neural systems are implemented using techniques inspired by greater understanding of how real neurons work. Computational Neuroscience has made considerable progress in recent years on spiking neuron based models of sensory and cognitive processes in the mammalian neo-cortex. Spiking Neural Networks (SNNs) are the 'third generation' of Neural Networks (Maass,

1997); the first generation being networks consisting of simple McCulloch-Pitts neurons (McCulloch and Pitts, 1943) with binary outputs and the second generation consisting of neurons with continuously-valued activation functions. Spiking neurons mimic how real neurons compute: with discrete pulses rather than a continuously varying activation. Depending upon the application and required level of biological detail, there are various types of spiking neuron model to choose from. However, there is also a trade-off between the level of biological detail and computational overhead (for a review and discussion see Izhikevich, 2004).

The spiking neuron is, of course, still an abstraction from an actual neuron, but a much more biologically plausible one especially as models can incorporate spike-timing based learning. Thorpe et al. (1996) presented experimental evidence for fast processing (occurring within 100ms of an image presentation) in the human visual system which implies that spike–timing information may be more important than spike rates as there is not enough time to generate a meaningful spike rate in very short time intervals.

Spike Timing Dependent Plasticity (STDP) is a currently favoured model for learning in real neurons. Experimental and modelling studies have shown that this form of Hebbian plasticity, where the relative firing times of pre and postsynaptic neurons influence the strengthening or weakening of connections, is the mechanism that real neurons use (Song et al., 2000). When firing times are causally related (i.e. the presynaptic spike is emitted before the postsynaptic spike) then the synapse is strengthened (Long Term Potentiation or LTP). When firing times are not causally related (i.e. the postsynaptic spike occurs before the presynaptic one) then the synapse is weakened (Long Term Depression or LTD).

Advances in software and hardware over the last ten years or so have made SNNs an increasingly feasible option for robotics applications. On the software side several general purpose spiking neuron simulators are freely available which means that

researchers do not have to code a modelling framework from scratch, and they also benefit from a community of users using the same tool. Desktop computing hardware is now available that can perform parallel processing (e.g. GPU) at an affordable price. But this can only take us so far. The emerging field of Neuromorphic Engineering is making it possible to simulate large neural networks in hardware in real time. 'Neural chips' are massively parallel arrays of processors that can simulate thousands of neurons simultaneously in a fast, energy efficient way, thus making it possible to move neural applications on board robots. This technology is currently being employed in dedicated hardware devices to perform specific bio-inspired functions, for example, the asynchronous temporal contrast silicon retina (Delbruck, 2008) and the silicon cochlea (Chan et al., 2007). There have also been several larger-scale projects for general purpose brain modelling. For example, the CAVIAR project; a massively parallel hardware implementation of a spike-based sensing–processing–learning–actuating system inspired by the physiology of the nervous system (Serrano-Gotarredona et al., 2009), the FACETS project (completed in 2010) delivering both neuromorphic hardware and software (FACETS website, 2012), and the NeuroGrid project at Stanford which has developed a hybrid analogue-digital neuromorphic solution capable of modelling up to 1 million neurons (reviewed in Silver et al, 2007). More recently, the SpiNNaker project has delivered a state-of-the-art real-time neuromorphic modelling environment that can be scaled-up to model up to a billion point-neuron models (Jin et al. 2010). The Spinnaker project aims to use this technology to investigate both how massively parallel computing resources can accelerate our understanding of brain function and also how our growing understanding of brain function can point the way to more efficient parallel, fault-tolerant computation.

These recent parallel advances in computational neuroscience and in the hardware implementation of large-scale neural networks, provide the opportunity for an

accelerated understanding of brain functions and for the design of interactive robotic systems based on brain-inspired control systems.

However, currently there are very few practical robotics implementations using neuromorphic systems. Two notable works are Linares-Barranco et al. (2007) which developed a solution using both a silicon retina, an FPGA and neuromorphic hardware to enable a humanoid robot to point in the direction of a moving object, and, more recently Davies et al. (2010) which developed a line following robot using a silicon retina and a prototype 4-chip SpiNNaker neuromorphic board.

More work needs to be done to develop practical applications that have a solid biologically-inspired theoretical basis and which can be scaled up and transferred seamlessly to run on neuromorphic hardware. For realistically large and effective SNNs to become possible in robotic hardware, ensuring that future neural models and simulations are actually implementable in neuromorphic hardware is important. It is also important to develop models which challenge the capabilities of such hardware and stimulate further developments.

The present project aims to contribute to this research area by investigating principles from computational neuroscience research to develop systems that can realistically be implemented in neuromorphic hardware for real robots in realistic environments: in particular to develop capabilities inspired by how nature manages to solve sensorimotor coordination tasks in an efficient way and to take advantage of some of the desirable features of real neural systems: low power consumption and real time operation but with flexible learning. The overall aim of the project is to build a self-organising, reflexive sensorimotor system using biologically inspired techniques based upon human cortical development.

The current work takes a developmental approach from the point of view that natural systems do not spring into being fully formed but undergo considerable periods of

refinement and change, and in particular adapt in response to input from the environment. This crucial developmental process in both pre- and early post natal life sets the stage for later capability. Developmental approaches are not new to robotics and they are a main feature of the existing fields of Epigenetic Robotics and also to some extent Evolutionary Robotics. The importance of the developmental process, and in particular the gradual acquisition of capability, for robotics has been pointed out by previous researchers. For example, Gómez et al. (2004) concluded that an initial low resolution sensorimotor system which undergoes a developmental learning phase actually learns faster than when starting with a higher resolution.

In the current work, the developmental process is seen as three phases which can be summarised as follows:

1. **Activity-Independent** - establishment of an initial network
2. **Activity-Dependent –** the first stage refinement of the network leading to organised, topological maps
3. **Lifelong Learning** – later stage refinements where maps learn to coordinate to build up useful skills

Figure 1 gives a graphical overview of the stages.

Activity Independent development corresponds to an initial phase where neurons and connections between them are set up. The maps are random and unorganised at this stage. Activity Dependent development is an analogue of the early post-natal stage of synaptogenesis followed by pruning in the real human cortex. It involves significant reorganisation and refinement of the initial networks to develop individual visual and motor topographic maps. A very important aspect of this stage is that it is activity dependent: as in real biological development, the quality and variety of stimuli in the early stages of development will determine later capability. The last developmental phase represents the beginning of ‘lifelong’ learning where useful skills begin to be

acquired. Here the individual cortical maps can form associations to coordinate visual and motor experience. The aim of the current work has been to build a generic capability for visuomotor coordination rather than try to achieve a task-specific scenario. It concentrates on how direction of movement of objects in the visual field and direction of movements of the robot's own body can be independently encoded into separate motor and visual maps, which, in later stages can be coupled to achieve a basic form of visuomotor coordination. Although the emphasis here is specifically on integrating visual and motor modalities, the techniques are generic enough to be used for other modalities as well.

The underpinning concept of the work is the Self-Organising Feature Map (SOFM) which is an analogue of how biological brains manage to represent complex multidimensional information from their environment as a 2D map in the cortex. Experimental and computational modelling work, in particular with visual systems, has shown that neurons in the cortex naturally form 2D maps as a representation of many-dimensional input information from the environment. (Dayan and Abbott, 2001). These maps are often called 'feature maps' as collections of neurons are specialised to detect certain features in an input signal (Grossberg, 1975). Such cortical maps are self-organising in that they primarily develop their structure according to the input information they need to represent and the pattern of connectivity between neurons is *activity dependent.*

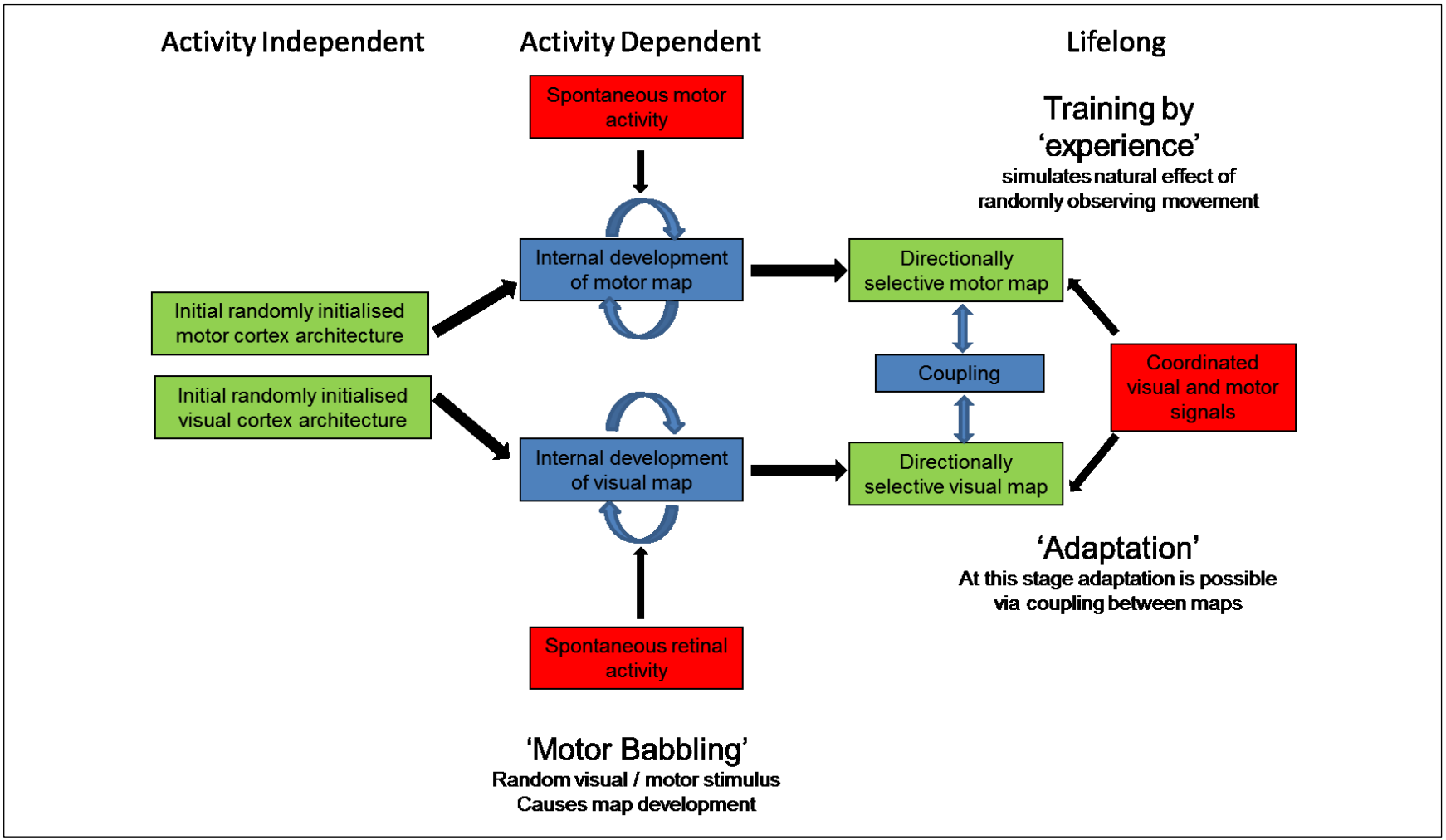


Figure 1- An Overview of the Developmental Stages

In the current work, the SOFM methodology is inspired by the Kohonen Self Organising Map (SOM) (Kohonen, 1995). The original Kohonen SOM is an unsupervised learning technique most commonly used for machine learning, for example, high-dimensional data clustering applications.

SOMs have previously been used as abstract models of the mammalian cortex and much work has been done specifically with implementations of the visual cortex and also some sensory-motor tasks such as visuo-motor control (Willshaw and von der Malsburg, 1976; Ritter et al., 1989; Goodhill, 1993; Miikkulainen et al., 1998; Metta et al., 1999). However, there have been relatively few implementations using spiking neurons. Of those that do exist, some use spike-rate based forms of learning (for example Choe and Miikkulainen, 1998) and some use spike-timing based methods (for example, Ruf and Schmitt, 1998; Sala et al.,1998; Panchev and Wermter, 2001; Bohte et al., 2002; Marian, 2002; Alamdari, 2005; Pham et al., 2006). Although there have been several applications of SOMs to robotics tasks (for example, Krose and Eecen, 1994; Terada et al., 1998; Toussaint, 2004; Toussaint, 2006), biologically-inspired implementations using spiking neurons and spike-timing based learning are scarce. One example is the work of Alamdari (2005) who used a self-organising spiking neural network with spike-timing based learning rules for robot path planning. Although SOM theory would seem to be a good choice for implementing a self-organising sensorimotor controller based upon real cortical processes, there are three main drawbacks to using this technique in autonomous robotics applications; all related to the fact that the process needs to be self-regulating as the robot needs to learn from the information available in its environment. Firstly, SOM network development is generally thought of in terms of two distinct phases: initial topological ordering followed by weight convergence and in computational models the phases are usually managed explicitly by the manipulation of neighbourhood size and learning rate parameters. Both of these parameters are normally

systematically reduced in a non-linear fashion during the course of training according to predefined schedules. In the case of an adaptive sensorimotor controller for a robot it is not ideal to have to predefine such schedules to control map development, instead the development should be self-controlling as it is in natural systems. Secondly, SOMs are very commonly used for high-dimensional data clustering applications due to their dimension reducing properties. Usually a long training phase is used to guarantee a precisely ordered map for the purpose of accurately representing the input data. The amount of training data is also usually defined in advance, either by generating it from existing empirical datasets or by creating specific training datasets. For a robot sensorimotor controller, a system that can regulate itself according to the input provided by its environment and determine for itself when the map is trained is much more desirable. Several previous works have created robotic learning systems where the data are generated 'online' from an environment (Krose and Eecen, 1994; Terada et al., 1998; Toussaint, 2004, Alamdari, 2005, Toussaint, 2006). Thirdly, traditional SOMs are not usually designed to be adaptive to changing input. For conventional data clustering applications the aim is to fully train the map with predefined input and the map structure after training is completely fixed for the purpose of classifying the data it was trained with. In the case of cortical modelling, the maps are also trained for a fixed purpose to represent some aspect of experimental findings (for example the development of ocular dominance stripes). Furthermore, if a SOM trained with standard techniques is then trained with different data it often experiences 'catastrophic forgetting' – i.e. previous map organisation is overwritten. For robotic systems it is desirable to have behaviour more like the real cortex where there is some balance between remembering previous associations but also being able to adapt to new input. One particular aspect of real cortical learning that may be very relevant but is usually omitted in both traditional SOM and cortical modelling applications is structural plasticity: synaptic creation and

pruning. There is experimental evidence that in real systems both functional plasticity (synaptic weight changes) and structural plasticity (synaptic creation and pruning) may work together both in early cortical development and in the adult brain (Chklovskii et al., 2004; Butz et al., 2009; Gilbert and Li, 2012). It is possible that structural plasticity may provide the means to create better adaptive learning in robotics applications.

In summary, this research project has developed a software framework for creating visual and motor cortical maps using spiking neuron implementations of self-organising feature maps. Adaptations to traditional SOM techniques have been made to address the majority of the issues mentioned in the previous paragraph, especially with respect to autonomous control of the training process. The framework also includes an adaptive coupling mechanism, also based on bio-inspired principles, which enables the association of separate visual and motor maps so that the visual input can control and modulate the motor response. The software implementations have been designed with a view to future implementation in neuromorphic hardware, in particular the SpiNNaker system (Jin et al., 2010), which in the future will be deployed on a medium-sized humanoid robot to provide basic visuomotor coordination for tracking a moving object. Another novel aspect of this research work is the use of the DVS 128 Silicon Retina camera (a specialised neuromorphic device) in the training of the visual directionally selective cortical map.

1.2 Research Contributions

The major contributions that this research delivers are as follows:

1. Previous research with self-organising maps has been extended to create a biologically-inspired developmental framework which encompasses both motor and visual cortical map creation as well as allowing coupling between the two types of map to achieve visuomotor coordination.

2. A methodology has been developed that enables internal regulation of cortical

map development and dispenses with predefining traditional aspects of SOM training such as learning schedules, neighbourhood reduction schedules and the amount of training data.

3. A methodology has been developed to use the DVS 128 Silicon Retina camera as input to train a directionally selective visual map.

Several parts of the research described in this thesis have already been published, or are currently submitted for publication in peer-reviewed journals. See the declaration on page xii for details and copies of the publications at the end of the thesis.

1.3 Thesis Structure

This thesis is divided broadly into four parts. Part I serves as the introduction to the work and includes Chapters 1 and 2. Part II contains the core theory and implementation details of the various elements of cortical feature map modelling and training and includes Chapters 3-5. Part III looks at specific extensions of the theory and methods in Part II with respect to robotics applications and 'lifelong learning' and includes Chapters 6 and 7. Part IV presents details of the experiments performed and results obtained for each stage of implementation and the final discussion and conclusions. It includes Chapters 8-11.

A chapter-by-chapter summary is given below:

Chapter 2: Motivation and Background

The main literature review of topics relevant to this research. Concludes with a summary of the main works which have provided inspiration and where they have been extended and improved upon.

Chapter 3: A Motor Cortical Feature Map

Describes the theory and methods used in creating and training a directionally-selective motor map.

Chapter 4: A Visual Cortical Feature Map

Describes the theory and methods used in creating and training a directionally-selective visual map.

Chapter 5: Adaptive Plasticity

Describes the theory and methods used to modify traditional SOM training to dispense with predefined parameters such as the learning rate and to make the training process autonomous and activity dependent.

Chapter 6: Sensorimotor Integration

Presents an exploration of neural-motor integration in simplified environments in lieu of implementation on a real robot. In the first case the implementation of a simplified animal sensorimotor behaviour is described. Secondly, a simulated humanoid robot implementation using the motor map theory from Chapter 3 is described.

Chapter 7: Coupled Maps

Describes the methods by which the visual and motor maps are coupled together.

Chapter 8: Cortical Feature Map Training

Presents the experiments and results for the Motor and Visual map training

Chapter 9: Training with Adaptive Plasticity

Presents the experiments and results for Motor and Visual map training including the Adaptive Plasticity methods of Chapter 5.

Chapter 10: Coupled Map Training

Presents the experiments and results for the coupling of the visual and motor maps.

Chapter 11: Discussion

The final discussion of the results and conclusions of the research as well as future work.

2 Motivation and Background

2.1 Introduction

This chapter covers the background literature which has provided motivation for the various elements of this research work. The following sections contain: a brief discussion of basic spiking neuron theory and options for simulation which have informed the choice of neuron model and simulator for this research work, a review of previous research works in robotics that have used theory from Computational Neuroscience, an overview of traditional Self-Organising Map theory including previous works that have adapted its principles for cortical modelling and finally a review of previous works that have looked particularly at sensorimotor applications. The chapter concludes with a summary of which elements have inspired the current work and what advances on previous work are proposed.

2.2 Spiking Neural Networks and Simulators

The Spiking Neuron

Artificial Neural Networks (ANNs) are now a common technique used in many computing applications. Maass (1997) distinguishes three generations of ANNs. The first is the simple threshold neuron of McCulloch and Pitts (McCulloch and Pitts, 1943) which can output binary signals and are universal for digital computations. The second generation improves upon the McCulloch and Pitts neuron by using an activation function allowing for continuously varying output and these are universal for analog computation. The third generation is the Spiking Neural Network (SNN) in which artificial neurons compute with pulses, much like real neurons do. As early as 1907, Lapique developed a model of a neuron which formed the basis of the Leaky Integrate and Fire neuron (LIF) which is still widely used today (Lapique, 1907; Abbott, 1999). The LIF model captures the essential behaviour of a spiking neuron: contributions from

connecting neurons increase the membrane voltage of a target neuron until the voltage reaches a threshold value and an action potential or spike is generated. The membrane voltage is then reset to a specified value and there is a refractory period during which the neuron cannot fire. In the absence of further action potentials the membrane voltage relaxes exponentially to a resting potential (Dayan and Abbott, 2001). Figure 2 shows the behaviour of a typical simulated LIF neuron with a threshold potential of -40mV, reset potential of -70mV and refractory period of 5ms. When many spiking neurons are combined together they can be used to study the properties of large neural networks and the implications of large numbers of synaptic connections in such networks (Abbott, 1999).

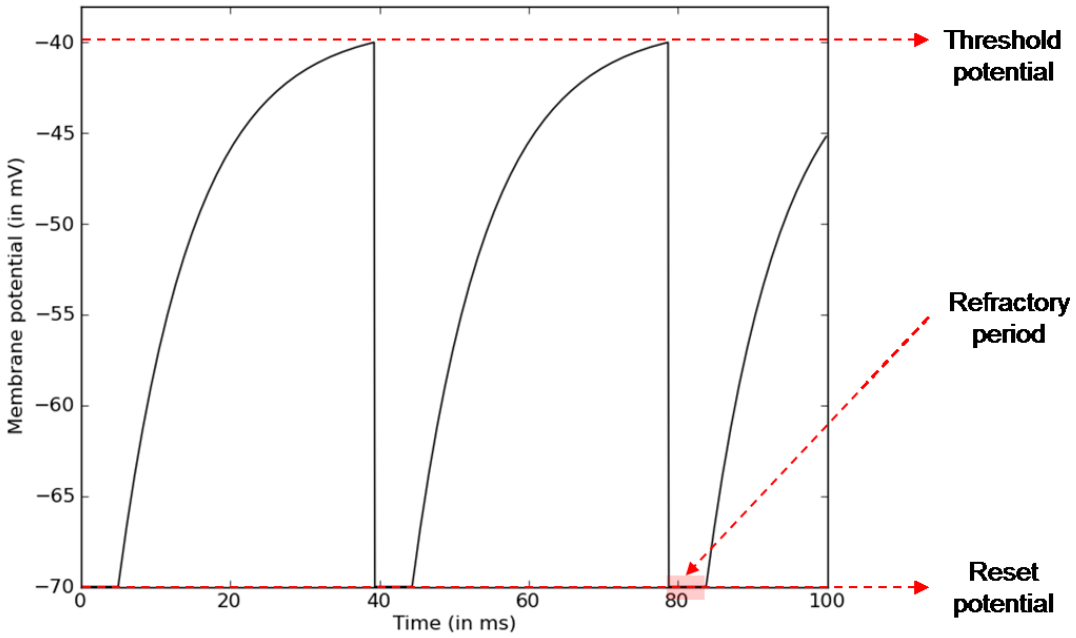


Figure 2- The Leaky Integrate and Fire Neuron

There are several options for information coding in spiking neural networks which are reviewed in Gerstner (1999). Basically these can be viewed as *rate codes or pulse codes* (although the distinction is not completely clear cut in some cases). Rate codes have been very popular and can indeed explain a lot of experimental results. The traditional neural network models of the first and second generation are effectively rate based.

However, drawbacks of this approach when considering biological plausibility are that it is assumed there is a long enough period of time for a meaningful rate to be defined and/or a population which acts homogenously (effectively that each neuron behaves in a similar way independent from the action of other neurons in the population) over which a rate can be taken. Experimental findings from visual object recognition tasks and some motor behaviours have shown that the response is much too fast for a rate coding scheme to have encoded the information (Thorpe et al., 2001). As an alternative, pulse coding (or spike-time encoding) is closer to how real neurons compute and allows many more options for encoding with spike times ranging from the time of individual neuron spikes, variations between the times of a group of neurons, and synchrony of firing between neurons. Another point in favour of using a spike-time coding method is that it has been established that spike timing is important in the mechanism of learning (Bi and Poo, 1998; Song et al, 2000; Froemke and Dan, 2002).

When implementing an SNN there is a choice of neuron models depending upon the type of biological properties one requires. The table shown in Figure 2 of Izhikivich (2004) gives a summary of the types of model available, their properties and an indication of the computational overhead. At one extreme is the Hodgkin-Huxley model which can represent many biological properties as it models down to the level of ion channels, but is computationally expensive. At the other end the simple Leaky Integrate and Fire (LIF) neuron model is computationally cheap but can only represent the most basic spiking behaviour. Izhikevich has proposed a model which combines the possibility of a rich range of behaviour with efficient computation (Izhikevich, 2003).

Spiking Neuron Simulation

It was noted in Chapter 1 that today there are several general purpose spiking neural network simulators available to researchers, and as is the case with choosing a neuron model, there is a trade-off: in this case speed of simulation versus functionality and

ease-of-use. ModelDB is a central, online repository for storing Computational Neuroscience models which can be searched by model type and also simulator type (Hines et al., 2004). A quick examination of the current models (as of June 2012) shows that a large proportion use bespoke simulations coded by the authors in languages such as C++, Java, Python and MATLAB but an equal number of models use general-purpose simulators of which the most well-known are NEURON, NEST, PCSIM and Brian. The advantages of coding a simulation from scratch (say in C++) are that it is possible to code only the relevant functionality and such a simulation can be optimised to run as fast as possible. The disadvantage is that the researcher needs to write the code, unless he or she is lucky enough to find an existing solution that can be adapted. The advantage of using a general-purpose simulator is that one benefits from an existing framework and functionality which can be used to quickly construct models. In addition, there is (hopefully!) documentation, examples and support available. However, the learning curve required to master the features of a particular simulator needs to be taken into consideration. It is also likely that generic simulators may be less efficient in terms of run-time and memory usage as they have to include a large amount of functionality. From the point of view of applying Computational Neuroscience techniques to Robotics research, there are two other important factors which influence the choice of simulator. Firstly, if it is intended that the networks will be deployed to neuromorphic hardware at some point then it makes sense to (as far as possible) avoid the need for porting the code to a different language/simulator. Secondly, to consider what other functionality besides neural modelling might be required and how easy is it to integrate. For example, in a robotics scenario a physics simulation and visualisation of a robot may be useful as well as a means to communicate with real robot hardware.

2.3 Computational Neuroscience in Robotics

Arbib et al. (2008) define the field of Neurorobotics as

'*the design of computational structures for robots inspired by the study of the nervous system of humans and other animals*'

Over the last twenty years or so there has been extensive work with bio-inspired 'whole animal' models of behaviour inspired by specific invertebrate and vertebrate organisms (for example Arbib and Liaw, 1995; Damper and French, 2003; Meyer et al., 2005; Alnajjar and Murase, 2008). This type of modelling falls under the umbrella of Computational Neuroethology which encompasses the modelling of real animal behaviour grounded in biologically realistic neural models (Cliff, 1991, Arbib et al., 2008). An important component of Computational Neuroethology is to model situations where entire animal 'behaviours' are generated from interaction with the environment. In the words of Cliff (1991):

'*closing the external feedback loop from motor output and sensory input*'

There have also been research works not based upon any particular animal's behaviour but tackling generic concepts such as control and vision (for example Hagras et al, 2004; Wang et al., 2008; Stratton et al., 2009). The works which have used spiking neural networks manage to achieve robust behaviours and learning even with very simple architectures: for example, the Aplysia models of Damper and French (2003) and Alnajjar and Murase (2008). Such models are extremely useful as first steps in Neurorobotics research as they can provide insights into how nature has equipped animals with efficient survival strategies and moreover how it is possible to generate fairly complex behaviours with minimal neural architectures. Furthermore, actually implementing neuroscientific models onto robotic hardware enables validation of the biological theory in an embodied way (Arbib et al., 2008).

There have been a few research projects aimed at creating more complex models and robotic implementations based upon human-like capabilities using spiking neural

networks. Two notable examples are the Darwin series of robots (Krichmar and Edelman, 2003; Edelman, 2007) and the humanoid CRONOS/SIMNOS project (Gamez et al., 2006; Gamez, 2008). To date, there appears to be little or no work using spiking neural networks to generate bio-inspired behaviours for autonomous small to medium size humanoid robots.

In summary, it is only relatively recently that works using spiking neural networks for practical robotics applications have begun to emerge. This is most likely due to two main factors. Firstly, the type of software and hardware required to effectively implement spiking neural systems for autonomous robots has only recently become more accessible to robotics researchers. Secondly it has been necessary to wait for the required advances in Computational Neuroscience theory and modelling to become available for transfer across to the robotics field of research.

2.4 Modelling Cortical Feature Maps

Biological Feature Maps

Through computational modelling work, and validation from neuroscientific experiments it is accepted that neurons in mammalian cortex naturally form 2D maps as a representation of information received from the environment. These maps consist of distributed but often overlapping populations of neurons that respond preferentially to features in an input signal. Cortical maps are *self-organising:* they develop according to the information in the input they need to represent in an activity-dependent manner (Dayan and Abbott, 2001). An important feature of these maps is that they are dimension-reducing as they represent a range of many-dimensional salient information from the environment as a 2D map. These maps are often referred to as 'cortical feature maps' or Self-Organising Feature Maps (SOFMs). The most well-known maps are from the visual system, for example:

- Ocular dominance (OD) – neurons respond to input preferentially from either the left or right eye and groups of neurons with the same dominance properties form distinct bands or stripes (Hubel and Wiesel, 1977). Figure 3 shows part of an experimentally determined OD map.

Figure 3 has been removed due to Copyright Restrictions

Figure 3- Ocular Dominance Map from Macaque Visual Cortex

(taken from Fig 24(a) in Hubel and Wiesel,1977)

- Orientation Selectivity (OS) – neurons are responsive to visual objects of a particular orientation and neurons with similar response properties are grouped together in cortical patches (Hubel and Wiesel, 1977). Figure 4 shows part of an experimentally determined OS map.

Figure 4 has been removed due to Copyright Restrictions

Figure 4- Orientation Selectivity Map from Shrew Visual Cortex

(adapted from Fig 1(B) in Bosking et al.,1997)

- Direction selectivity (DS) - neurons are responsive to visual objects moving in a particular direction and neurons with similar response properties are grouped together in cortical patches (Weliky et al., 1996). Figure 5 shows part of an experimentally determined DS map with arrows showing the preferred direction superimposed.

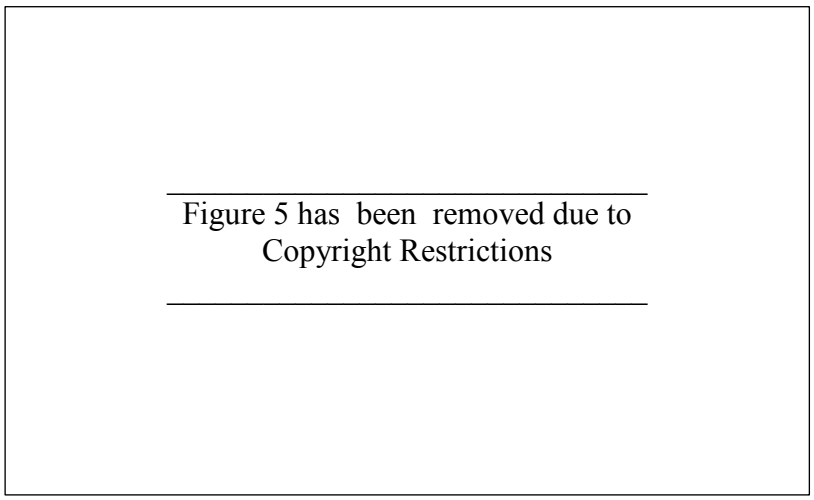


Figure 5- Directional Selectivity Map from Ferret Visual Cortex
(adapted from Fig 2(a) in Weliky et al.,1996)

As can be seen from the maps in Figure 4 and Figure 5 there is some overlap in response because the degree of selectivity or 'tuning' of individual neurons varies from being very specific to more broadly tuned.

The Self-Organising Map

The Kohonen Self Organising Map (SOM) is a well-known technique for modelling feature maps using an *unsupervised* learning process (Kohonen, 1995). A SOM network usually has two layers only: an input layer which passes in training data and does no processing and an output layer which forms the actual map. These two layers are usually fully interconnected. Figure 6 shows a typical SOM network arrangement. The input layer has as many nodes (neurons) as there are dimensions of data and normally the output layer is chosen to be of an appropriate size to represent the number of input patterns: in a traditional SOM one output node responds to a particular pattern so the output layer must contain at least as many neurons as there are different types of input

patterns. The Kohonen SOM learns to represent the range of input data available and in the final map the data is *topologically* arranged (similar inputs are mapped to similar spatial locations in the map). The weights between the input and output nodes store the information elements represented in an input pattern vector. Consequently, the number of connections to a neuron/node determines the maximum dimensionality of the map.

Figure 6 has been removed due to Copyright Restrictions

Figure 6- The SOM Network Architecture

(taken from the SDL Component Suite website, 2008)

The SOM training process is as follows:

1. Present an input vector (pattern) to the input layer
2. Select the winning output layer node (the node with the highest activation)
3. Determine a spatial neighbourhood around the winning node
4. Adjust only the weights of nodes within the neighbourhood
5. Decrease the neighbourhood size (N), and Kohonen learning constant (k)
6. Repeat 1…5 until map has converged

The weight adjustments at step 4 are calculated using the Kohonen learning Equation (1).

$$\Delta w_{ij} = k(x_i - w_{ij}) \quad (1)$$

Where:

w_{ij} is the weight between input node i and output node j

x_i is the value of the input vector element applied to input node i

k is the Kohonen learning constant

Note that in the traditional SOM method described above, there is implicit activity dependence: the 'winner take all' strategy for learning in step 2 ensures that only one output unit is active for a particular input pattern. Also, nodes whose weights never cause them to be activated because they do not match any input data closely enough are never reinforced and are thus redundant.

In step 5 the decrease of the neighbourhood size N and the learning constant, k is normally not done linearly as map creation is thought of as having two phases which operate at different rates. Firstly, an initial topological ordering where there is the greatest amount of change and secondly, weight convergence where there is less disruption as the map is settling into its desired structure. There are various options for non-linear reduction rules, for example, simple exponential decay. Mulier and Cherkassky (1994) challenged the efficacy of traditional approaches to reduction schedules for learning rate. Their experiments found that the choice of learning rate in a traditional SOM can drastically affect how the data contributes to the ordering of the final map – in effect they found that up to 80% of the data could be wasted. This effect can be a particular problem when training data is 'recycled' (i.e. presented more than once) and even random presentation of training samples does not completely get rid of the problem. They proposed a new, statistically-based learning rate function which

ensures that each training pattern contributes equally to map formation. They also stated that the neighbourhood function has a contributory effect and the learning rate function they propose is independent of it. Keith-Magee et al. (1999) emphasised the importance of neighbourhood reduction and proposed a specific two-stage reduction scheme where the neighbourhood size is reduced by a factor of n/2 for n training epochs and thereafter reduced by R, the resolving distance of the SOM which is the distance between peaks if every data point was distributed evenly over the map. Surprisingly, there is actually some evidence from previous work in generic SOM theory that reducing the neighbourhood size during training may not be crucial for map formation at all: the work of Raginsky and Anastasio (2008) demonstrated that the optimum neighbourhood size for best representation of input information is finite and small. Research works that have performed practical experiments which support this view include Alamdari (2005) and Pham et al. (2006). Both used a fixed, small Gaussian neighbourhood successfully in SOM implementations.

The issue of defining learning rate and neighbourhood parameter reduction in relation to traditional SOMs has been noted by several previous researchers. For example, Berglund and Sitte (2006) and more recently in Berglund (2010) the Parameter-Less SOM or PLSOM is described. These works developed a method of controlling the learning in a SOM by using the ratio of the last error between the input vector and weight vector of the winning node to the largest previous error as a scaling factor (Berglund and Sitte, 2006). In later improvements, the ratio of the last error to the diameter of the input space is used (Berglund, 2010). Shah-Hosseini and Safabakhsh (2000; 2001) developed the TASOM or Time-Adaptive SOM. Here, each neuron has its own learning rate and neighbourhood parameters and these are changed according to the distance measure between the current input vector and the synaptic weight vector of the neuron. More recently, Shah-Hosseini (2011) has developed a variant called the Binary

Tree TASOM, which incorporates the removal and addition of neurons during training to allow adaptation in an environment where the inputs can change. Brohan et al. (2010) tackled the problem in a slightly different way by allowing the learning rate and neighbourhood parameters to reset to their initial values based upon the novelty of the input, but otherwise using linear reduction schemes for both.

In step 6 of the Kohonen method, the training process is repeated until the map has converged. Miyoshi (2005; 2007; 2008) discusses some of the conventional methods for determining when a map has converged: usually either when the number of training cycles exceeds a threshold, or when the largest distance in all distances between learning data and its winning node becomes smaller than a threshold. Miyoshi proposes that both of these are somewhat deficient as they do not consider the actual learning rate or look at the convergence of a large enough amount of nodes. Miyoshi proposed a new method which keeps a track of the sum of 'distance' changes for all weights of all patterns in a training cycle. It is assumed that this should become constant when convergence is achieved.

Cortical Feature Map Modelling

This section looks at some previous research works over the last 20 years or so which have created computational models reproducing features of real cortical feature maps, and are relevant to the goal of modeling cortical feature maps with flexible, efficient learning.

Goodhill (1993) proposed the first computational model of retinotopic map formation in visual cortex where Ocular Dominance stripes developed in the presence of correlated input from two 'eyes' without adding any special assumptions. The map learning mechanism is inspired by Kohonen (1982) and includes a winner take all mechanism whereby lateral inhibition suppresses the activity of all neurons in an area apart from the maximum responder. The maximum responder and its neighbourhood have their

weights increased by a Hebbian learning rule. Normalisation of weights is also required to ensure they do not increase unbounded. Following their proposed model of Spike Timing Dependent Learning (STDP) (Song et al., 2000), Song and Abbot applied STDP theory to generic cortical map development to establish if STDP alone can account for map development (Song and Abbott, 2001). They looked at various scenarios with learning on feedforward connections only, learning on feedforward and lateral connections and also the role of inhibitory lateral connections in map formation. Their experiments showed that STDP could indeed provide the competition mechanism between synapses needed for map development and furthermore that lateral excitatory and inhibitory connectivity were required for the map to be refined and preserved. This work was a significant advance in that it demonstrated that it was possible to model map development in a similar fashion to a Kohonen SOM but with spiking neurons and a biologically inspired learning method. More recently Dayan (2004) has proposed a model of Ocular Dominance stripe formation based upon activity dependent mechanisms and including a special 'arbor function' which controls the setup of the initial, unrefined map. Essentially the arbor function imposes a basic topography on the initial map. This model includes a simple Hebbian learning rule and normalization. Shon et al (2004) explored how directionally selective (DS) maps could form using STDP. Confirming the earlier results of Song and Abbott (2001) they found that lateral connections are vital for the development of maps and that the combination of lateral excitatory and inhibitory connectivity is required for robust directional selectivity. Their work used a LIF neuron model and 'retinal' input consisting of moving bars; the retinal activity was passed through a set of filters reproducing the ON/OFF activity of the Lateral Geniculate Nucleus (LGN). Their STDP method incorporated an asymmetric STDP time window which was important in the development of directional selectivity. A similar work looking at the development of directional selectivity in the Primary

Visual Cortex using STDP, Wenisch et al. (2005) also proposed that an asymmetric STDP time window is important and show that directional selectivity can arise by STDP even with only intrinsic spontaneous spiking activity. Unlike earlier works, they also consider the role of neuronal delays on the lateral connections (the larger the separation between pre and post neurons, the longer the signal takes to integrate) and find that these are possibly essential in the detection of moving objects. Unlike Song and Abbott (2001) and Shon et al. (2004) STDP learning is only on the lateral connections and feedforward connections play no role other than the relaying of the input data.

The LISSOM (Laterally Interconnected Synergetically Self-Organizing Map) framework has been used to model many features of visual map development (for example, Bednar and Miikkulainen, 2000, 2003; Miikkulainen et al., 2005). These models incorporate learning on both afferent and lateral connections and through various experiments they demonstrate the important role of lateral connections in initial map development and adaptation during recovery from lesion. For example, the RLISSOM (Reduced LISSOM) variant is used to demonstrate how lateral connections are responsible for reorganisation in the visual cortex after a retinal lesion. More recently, Gilbert and Li (2012) have reviewed current thinking on plasticity in the adult visual system and support the view that adult perceptual learning and recovery after lesion may share the same underlying mechanism involving changes in lateral connections. The PGLISSOM (Perceptual Grouping LISSOM) variant is used to demonstrate the role of lateral connections in perceptual grouping. Unusually, for current thinking at the time of this work, the LISSOM models allow plasticity on inhibitory as well as excitatory connections. In most previous works inhibitory plasticity was rare due to a lack of strong experimental evidence for learning on inhibitory connections in the cortex. In LISSOM, Mikkulainen et al. (2005) justified it on the grounds of a modelling abstraction: an inhibitory connection is taken to incorporate a

plastic excitatory connection onto an inhibitory interneuron. The LISSOM models demonstrated the important potential role of lateral inhibitory connections in providing competition in the response of the map: the long–range inhibitory connections suppress the activity of neurons in remote areas when a particular area of the map is active; hence neurons which develop similar response preferences are spatially collocated. The PGLISSOM model showed how the inhibitory connections perform the role of segmentation in perceptual grouping experiments. Bednar and Miikkulainen (2003) looked very specifically at the co-formation of both directionally selective (DS) and orientation selective (OS) maps. The HLISSOM variant used in these experiments included a specific model of LGN processing with a set of 4 layers in a temporally delayed hierarchy for both the ON and the OFF LGN components. Their conclusions were that a single map organisation system could be used for the development of two different visual maps and that the delayed LGN input was important for the development of directional selectivity.

In a very recent work, Honda et al. (2011) investigated the mechanism responsible for the development of direction selectivity in the Xenopus (frog) retinotectal system using a neural circuit model with STDP**.** Their model used both feedforward and lateral learning and their conclusions emphasise the role of delayed feedforward inhibition. In their discussion they acknowledge that many previous works have developed successful directional selectivity models using different mechanisms!

2.5 The Sensorimotor Loop

This section looks at some previous research works which have implemented sensorimotor learning and coordination systems. In particular, a review of previous works specifically related to the use of self-organising or developmental principles for robotics, either simulated or using actual robot hardware.

Ritter et al. (1989) applied Kohonen SOM theory to visuo-motor coordination for a

simulated robot arm. Prior to the application described in this paper there had only been theoretical work done on motor maps for robot control. Furthermore, work on motor maps in general was scarce with the majority of self-organising map studies being for visual systems as described in Section 2.4. In Ritter et al. (1989) the map is learned from a random sequence of arm movements which are observed by a camera. Two methods are used: learning arm kinematics (positioning the end effector at a specific location) and learning arm dynamics (regulating torque and speed for ballistic movements). This method is different from a standard SOM due to the fact that there are actually two maps: one from input to output layer and one from output layer to motor command. Training the network using arm kinematics proceeds by presenting a random location and selecting the winner neuron. A rough set of output joint angles are calculated which move the effector to roughly the correct vicinity. The output values are refined and a 'fine' movement is used to position the effector accurately. Approximately 4000 learning steps were required to get a fairly accurate result and 20,000 learning steps to fully capture the desired input-output mapping. For learning arm dynamics the structure of the network is the same, but the output is a triple of torque values. Krose and Eecen (1994) also made use of Kohonen SOM theory, but for environment representation in the sensory domain for robot navigation. Prior to this work, most previous applications of SOMs for path planning represented the world space, but in this work a SOM is used to map sensor space and is constructed by exploration. The map is a 3D lattice but trained using the regular Kohonen SOM methodology. Once the sensory map has been constructed a path planning algorithm can be used to move the robot from a given state to a goal state. In their conclusions the authors stated that one drawback of this approach is that many training iterations (approximately 100,000) were required to construct the SOM and create the transition probabilities for path planning. Terada et al. (1998) have also used SOM theory for sensory motor mapping in

robot navigation. Their VSAM architecture (Visual State Action Map) used raw image data to firstly train a visual SOM. Then reinforcement learning (Q learning) was used offline to associate output from visual map with appropriate motor action. The end result was that each SOM node had a list of possible actions and resulting next states (sub nodes and links to next state). Similarly to Krose and Eecen (1994) a disadvantage for this approach is the overhead for computing all the possible transition states during Q learning.

A significant advance over previous works was that of Metta et al. (1999) which looked at visually guided reaching, using an actual 10 Degree-of-Freedom robot arm/head setup. Of particular relevance to the rationale for the current work is the emphasis on applicability of biological developmental approaches to robotics and a discussion of why previous approaches have not resulted in robots with anything like human capabilities. Metta et al. (1999) perceived that previous attempts to model sensorimotor control have been based upon learning rather than development, and in their view biological systems do not spring into being as a 'blank slate' but instead undergo a developmental phase where capabilities develop within the context of an environment. One human developmental example they describe is the early presence of the Asymmetric Tonic Neck Reflex (ATNR) which plays a crucial role in allowing babies to see their hands and improve visual fixation of the hands (White, Castle and Held, 1964; Bushnell, 1981). The new-born's own body motions are the means to establish a coupling between perception and action. The methods described in Metta et al. (1999) are not traditional SOM but instead based upon the 'motor primitives' work of Bizzi, Mussa-Ivaldi and collaborators (Mussa-Ivaldi, 1992; Mussa-Ivaldi and Giszter, 1992; Mussa-Ivaldi et al., 1994; Bizzi et al., 1995). A motor primitive can be described as a torque field which controls and coordinates one joint and motor primitives can be combined to move more than one joint simultaneously. In Metta et al. (1999) the goal is

for the system to learn which combinations of motor primitives place the arm at the correct location from the inputs of head control vectors and arm control vectors. Their experiments attempted to mimic early development of coordinated reaching by dividing the skill acquisition into two stages: firstly ballistic reaching to a static target is learned and secondly coordinated eye-hand movements towards moving targets are then required. The system copes with the latter even though it has not been trained on any moving target. It is also notable that only about 100 learning trials (5 minutes of real time) were required to get this performance.

Marian (2002) specifically used a spiking neuron variant of a traditional SOM, including a biologically-inspired spike-timing based learning rule, applied to a theoretical investigation of visuomotor coordination. Marian's work was directly inspired by the integrated Parieto-Frontal framework for visually-guided reaching proposed by Burnod and collaborators (Burnod et al., 1999). This work proposed that cortical control of reaching is distributed over a 'network of networks' in the parieto–frontal lobes. Although there are clearly distinct cortical areas for visual and motor processing, for tasks which require a combination of both (i.e. visually-guided reaching) there is no distinct area for the task. Burnod et al. proposed that coordination of the two cortical areas is achieved by recruiting populations of neurons from both areas. The Burnod model explains how the sensory-motor transformation is achieved using neural networks having a distributed representation of either position or direction and also how such networks can be combined to solve sensory-motor tasks. Since the publication of Burnod's framework, several experimental works support the existence of separate but interconnected networks in the cortex. For example, Pulvermüller (2005) and Boulenger et al. (2009) describe how speech related to actions produces activity not only in speech areas but also in the motor cortex pertaining to the relevant body parts. More recently, Archamboult et al. (2011) studied the activation of cortical neuronal populations in

monkeys during visually guided reaching. They confirmed a sequential activation of neurons in the premotor, motor and parietal cortex. Marian (2002) used a simplified interpretation of Burnod's theory to create a model of a coupled visual and motor map. The motor map was created using a spiking version of a SOM to develop directional selectivity. The development of visual directional selectivity was not modelled: instead a map was pre-created with directional selectivity already present. A separate coupling stage was then performed where connections between the two maps were refined using STDP finally allowing only visual input to generate a distinct motor response. Although a small-scale theoretical work, Marian (2002) is nonetheless important as it presented a biologically-inspired alternative to traditional methods of robotic control and solving the coordinate system problem. Since this time there have only been a handful of works which actually use the Burnod architecture in robotics applications (Carenzi et al., 2005; Zollo et al., 2005; Eskiizmirliler et al., 2005; Zollo et al., 2008). All of these are concerned with learning of initial 'modules' such as vision, proprioreception and then an accelerated second-stage learning which can combine the modules to learn new tasks such as reaching and grasping. Although the methods used in these works are bio-inspired, they do not employ spiking neural models.

Paine and Tani (2004) use a self-organising approach for navigation in a simulated wheeled robot based upon a hierarchy of Continuous Time Recurrent Neural Networks (CTRNNs). A lower layer learns basic motor primitives (in this case entire navigation actions such as turn left, go straight, etc.). A higher layer of control neurons develops a topologically ordered mapping of initial cell activation states to motor-primitive sequences and manage the selection of one primitive or another. The self-organising part of the system is provided by a Genetic Algorithm (GA) which is used to evolve the weights between layers.

In another innovative take on self-organising maps Kikuchi, Ogino and collaborators

use optic flow to enable soccer playing humanoid robots to learn mappings between visual stimuli and motion (Kikuchi et al., 2004; Ogino et al., 2005). The use of optic flow is important as it encodes information about the motion in a visual signal. Similarly to Paine and Tani (2004) there is the concept of a basic motion primitive which represents a movement option in the robot (for example, 'step left' , 'walk forward') but in this case encoded by an optic flow field. More complex 'motion actions' can be represented by a specific combination of the primitives (for example, 'approach ball'). The football behaviour 'passing the ball' is used as an example. This is decomposed into three component actions (approach ball, trap ball, kick ball) which are in turn decomposed into several motion primitives. The robots learn mappings for each primitive using a traditional SOM. To achieve the behavior the individual components need to be integrated and this is done by a simple set of rules, for example 'if ball is in front of foot then kick' and 'if ball is moving, trap it'.

Like Marian (2002), Alamdari (2005) also used a spiking neuron with spike-timing based learning in a SOM but applied to simulated robotic path planning. In this work STDP is used to adapt both synaptic weights and delays. The inputs are locations in the environment and the network learns the clusters forming obstacle-free locations. Path planning is implemented using a form of the A* algorithm modified to work with the network representation of the map.

Toussaint (2004; 2006) used a SOM applied for sensory motor mapping for a simulated robot navigating a maze. The architecture consisted of fixed motor and perceptual networks coupled to a growing SOM layer linking the two domains. Dynamic growing of the SOM layer is based on the Neural Gas (Fritzke, 1995) and FuzzyARTMAP (Carpenter et al., 1992) approaches. After 30,000 steps of exploration and learning a map which matches the structure of maze was obtained. Following this stage, planning was enabled and the robot given a goal position to navigate to. The planning dynamics

were based upon classical reinforcement learning and enabled the robot to learn to avoid blocked paths.

More recently Morse and Ziemke (2009) have used a hybrid model consisting of a hierarchical system of cortical maps constructed from Echo State Networks (ESNs) (Jaeger, 2002) and SOMs to model sensory-motor learning in a simulated robot task. The ESNs are used to model cortical dynamics without implementing the finer details of cortical structure. Input information from the environment is passed in via the ESN where the cortical dynamics are emulated. The ESN response is then classified using a SOM. The SOM also provides an input back to the ESN and the location of the winning SOM unit in SOM space is provided as output. The model is validated by replicating some of the experiments of Held and Hein (1967) in which kittens raised in the dark, and unable to control their own movements during exposure to light, exhibit severe deficiencies in visually guided behaviour, thereby confirming the importance of self-movement in acquiring visuomotor coordination.

2.6 Conclusions

This chapter has shown that although much work has been done in the individual domains of biologically-inspired Spiking Neural Networks and Self-Organising Cortical Feature maps the combined application of the two for robotic applications is still fairly sparse. The current work aims to integrate these aspects into one framework which can be used as the basis of a sensorimotor (visuomotor) controller for an autonomous robot. The following paragraphs give an overview of which ideas have been borrowed from previous work and where improvements or extensions have been made. Full details of methodology and techniques are given in chapters 3-7.

Methodologies

With respect to the potential use of SOMs in robotics projects, Chapter 1 raised some issues to do with making map creation as autonomous as possible, which are briefly

repeated here in relation to the existing literature discussed in this chapter.

Firstly, it is evident that previous researchers have recognised that it is important to investigate optimum ways to control SOM learning in terms of defining reduction schedules for the learning neighbourhood and rate, to try and get the fastest and best mapping of the input space possible (for example, Mulier and Cherkassky, 1994; Keith-Magee et al., 1999). Some researchers have also looked specifically at ways to control learning in a more adaptive fashion (Shah-Hosseini and Safabakhsh, 2000; 2001; Berglund and Sitte, 2006; Miyoshi, 2005; 2007; 2008; Berglund, 2010; Shah-Hosseini, 2011). However, none of these works has applied such ideas to a spiking neuron implementation of SOMs. Furthermore, the methods used in these works also assume that the SOM input is in the form of a pattern vector from which a meaningful distance to a weight vector can be calculated. The current research project has developed an approach to autonomous training of a spiking SOM which is biologically inspired and uses an adaptive method of controlling the learning rate which does not require a predefined learning rate schedule. This method works in the case of motor map development where the inputs are pattern vectors and also in visual map development where the input is individual spike events from a DVS 128 silicon retina camera. With respect to the learning neighbourhood, the approach used in Alamdari (2005) and Pham et al. (2006) is used: a small, fixed Gaussian neighbourhood is used throughout the training process and therefore a neighbourhood reduction schedule is not required.

Secondly, in traditional SOMs, the amount of training data is also usually defined in advance, either by generating it from existing empirical datasets or by creating specific training datasets. There are some robotic works where the data has been generated 'online' from an environment (for example, Krose and Eecen, 1994; Terada et al., 1998; Toussaint, 2004, Alamdari, 2005, Toussaint, 2006), and in the current work a method has been developed whereby the cortical maps are trained using inputs generated online

by random selection. Thus the composition and quantity of the input data is not defined and no assumptions are made about the amount of data needed to train a map. The adaptive learning mechanism described in the previous paragraph regulates the learning based upon the range and quantity of inputs seen and it is possible to use this mechanism to monitor whether a map is sufficiently trained or not.

Thirdly, traditional SOMs usually learn in a stationary or static environment (the range and quantity of input data is predefined). Furthermore, as the traditional learning rate and neighbourhood reduction schedules decrease monotonically, there is no possibility of undoing previous training. This is not an issue in the case of data classification applications or replication of specific experimental cortical feature maps as it is clear from the outset what the map is expected to represent. However, for robotic brain systems it is important to eventually have behaviour more like the real cortex where there is some balance between remembering previous associations and also being able to adapt to new input – i.e. to learn in a non-stationary or dynamic environment. The new adaptive learning mechanism developed in the current work contributes to solving this problem: the effective learning rate at any stage of the training process is determined in an activity-dependent way by the inputs seen so far. Therefore, if the input data composition changes, thus causing activity in new areas of the network, the learning rate can change to accommodate it. Adaptation requirements are also addressed in the section of work dealing with map coupling. As described in Chapter 1, the main aim of the current work has been to produce a system of a coupled visual and motor map for visuomotor coordination. The motor and visual maps develop independently, and then a ‘lifelong’ learning phase enables the maps to coordinate their activity. The coupling learning uses biologically-plausible STDP learning as well as a rewiring (synaptic creation and pruning) scheme designed to work with spiking neurons. As mentioned in the project rationale in Chapter 1, this is a neglected aspect of real cortical

learning in traditional and spiking SOMs and cortical modelling, but a possible means to endowing SOMs with greater adaptivity. Although the rewiring methods developed in this work are used only for coupling connectivity here, they are applicable generally to spiking SOMs.

From the discussion in section 2.4, Cortical Feature Map Modelling, it is clear that the majority of previous modelling work has been for visual system development, in particular Ocular Dominance (OD) and Orientation Selective (OS) maps. More relevant to the current work are models of Directionally Selective (DS) maps as for visuomotor coordination the visual objects will be moving. The work of Dayan (2004) and Shon et al. (2004) used 1D models with simplified visual input (e.g. manufactured Gaussian bars). The LISSOM models also included a variant for Directional Selectivity (HLISSOM; Bednar and Miikkulainen, 2003) which had a complicated architecture for ON/OFF processing and also used Gaussian bar input. The work of Wenisch et al. (2005) has inspired the visual system used in the current work as it is a 2D architecture, uses biologically plausible learning (STDP) and is a relatively simple setup (the LGN is not directly modelled). The current work also introduces a novel improvement over previous works by using visual input from a DVS 128 'Silicon Retina' camera. Therefore the input is real, not manufactured and in a biologically plausible form (spikes). There has been some similar previous work, for example Elliott and Kramer (2002) used a neuromorphic silicon retina chip to study the development of topography in the visual system, but in this case the silicon retina was small (16x16 pixels) and full map development was not modelled: only the development of retinotopy from Retinal Ganglion Cells to target cells. More recently, Galluppi et al. (2012) have used a DVS 128 camera as an input device to an orientation selective visual system implemented on SpiNNaker board.

In terms of sensorimotor applications, both theoretical and in practical robotics

applications, previous works have been mainly in the area of visually guided reaching. Marian (2002) created a simple, but much more general visuomotor framework based upon the theories of Burnod et al. (1999) but other similar work to date still concentrates on visually-guided reaching applications rather than generic frameworks (Carenzi et al., 2005; Zollo et al., 2005; Eskiizmirliler et al., 2005; Zollo et al. 2008). The current work uses the same approach as Marian (2002), however, the development of both motor and visual directional selectivity are modelled using a similar framework and for larger sized maps. The coupling of the maps is achieved by a method using STDP learning and also synaptogenesis and pruning.

Modelling

Based upon the discussion in Section 2.2, the Brian simulator has been adopted for the current work (Goodman and Brette, 2008). Brian is a general purpose spiking neuron simulator written in the Python[1] scripting language and has all of the most popular spiking neuron models built in plus the facility to define custom models using differential equations. The full source code is freely available and Brian can be easily customised and extended to interface with other Python packages. Currently there are many useful and freely available Python packages which include features such as physics simulation, visualisation, plotting and statistics. Brian can be installed on Windows, Mac and Linux and code created on one platform will run on others so it is a good choice of simulator to facilitate repeatability of experimental results. One disadvantage is that Python is not a compiled language, so with very large networks using STDP learning there will be an impact on run-time and memory usage. However, there are many options for optimisation in these cases which are well-explained in the

[1] http://www.python.org/

documentation on the Brian website[2]. Its capability has been found to be sufficient for the scale of modelling work done in this thesis. In terms of the potential for interfacing to neuromorphic hardware, the SpiNNaker system uses PyNN[3] as a 'front end' to develop applications to be deployed on the SpiNNaker hardware. PyNN is a simulator-independent language for building neuronal network models which is written in Python and integrates with many of the most popular simulators including Brian. Therefore it will be straightforward to easily convert pure Brian code to PyNN code and thence deploy to a SpiNNaker board.

[2] http://www.briansimulator.org/

[3] http://neuralensemble.org/trac/PyNN

3 A Motor Cortical feature Map

3.1 Introduction

This chapter describes the features of the spiking neuron motor map architecture and SOM training process used to create a directionally selective motor map. The methodology for the motor map is based primarily upon three previous works which have developed self-organising maps using spiking neural networks: Ruf and Schmitt (1998), Marian (2002) and Pham et al. (2006). Essentially, the aim is to create cortical-like maps which can self-organise to develop spatial and temporal selectivity to input patterns from an initially random state. Correlated (temporal and spatial) activity between neurons should strengthen both excitatory connections ('cooperation') and inhibitory connections ('competition') forming a topological map where neurons responding similarly to inputs are located together and those responding to different inputs are spatially separated. At this stage of the research the map development system does not have all of the desirable features mentioned in Chapters 1 and 2 with respect to adaptive learning. In the main only traditional SOM theory is used but applied to a spiking neuron scenario. Later improvements and additions for adaptive learning are left until Chapter 5.

3.3 The Neuron Model

A simple Leaky Integrate and Fire (LIF) model based upon the well-known CUBA (CUrrent BAsed) model described in Vogels and Abbott (2005) and Brette et al. (2007) is used and is described mathematically in equation (2).

$$\tau_m \frac{dV}{dt} = (g_e + g_i - V) \qquad (2)$$

Where:

V is the membrane voltage

g_e is the contribution from excitatory synapses

g_i is the contribution from inhibitory synapses

τ_m is the membrane time constant

Note that here the reversal potential is taken to be zero so is omitted from the model.

The neuron receives input from both excitatory and inhibitory synapses (represented by the terms g_e and g_i in equation 2) which are represented by the fast AMPA (α-amino-3-hydroxy-5-methyl-4-isoxazolepropionic acid) receptor model which assumes that the action potential generated by the presynaptic neuron is instantaneous and decays exponentially over time in between further action potentials (Dayan and Abbott, 2001). This behaviour is represented by equation (3).

$$\tau_s \frac{dg}{dt} = -g \tag{3}$$

Where:

g is the effective conductance for an excitatory or inhibitory synapse

τ_s is the synaptic time constant

When a presynaptic neuron fires the effective conductance for excitatory and inhibitory synapses (g) is updated as shown in equation (4).

$$g_{new} = g_{old} * w \tag{4}$$

Where:

g_{old} is the original effective synaptic conductance

g_{new} is the updated effective synaptic conductance

w is the synaptic weight

Table 1 gives a description of the neuron model parameters and their initialised values.

3.4 The Motor Map Architecture

The prototype version of the motor map network follows a typical SOM setup. It has a 16 neuron input layer and a 256 neuron output layer and is based directly upon that used in Marian (2002). Later in this chapter, section 3.7 describes how this network was scaled up to a larger size.

In the output layer 20% of the neurons are randomly assigned as inhibitory and 80% as excitatory as these appear to be the proportions of inhibitory to excitatory neurons in real cortex (Kandel et al., 2000).

Parameter	Value
Neuron Model	
V_{reset}, reset voltage	0 mV
$V_{Thresh,}$ neuron threshold	Randomly initialised as 3.9 mV plus noise normally distributed between 0 and 0.5 mV
τ_m, membrane time constant	5 ms
τ_s, excitatory/inhibitory synaptic time constant	5 ms
τ_{da}, delay on afferent synapses	2 ms
τ_{dl} , delay on lateral synapses	Set as distance between pre and postsynaptic neuron plus noise added by a Gaussian with mean 0 and standard deviation 0.5
τ_{refrac}, neuron refractory period	10 ms
Network Architecture	
N_{in}, number of neurons in input layer	16
N_{out}, number of neurons in output layer	256
w_{aff}, afferent synaptic weights	Randomly initialised between 0.4 and 0.5
W_{lat}, lateral synaptic weights	Randomly initialised between 0.3 and 0.4 (exc) and -0.3 and -0.4 (inh)
Exc_p_{conn}, connection probability for lateral excitatory connections	Calculated as *exp(-dist/sigma)* where dist is the Euclidean distance between the neurons and sigma is 3.5 (16x16 network) or 4.0 (48x48 network)
Inh_p_{conn}, connection probability for lateral inhibitory connections	Calculated as *exp(-sigma/dist)* where dist is the Euclidean distance between the neurons and sigma is 8.0 (16x16 network) or 15.0 (48x48 network)
Learning Rules	
η, the traditional SOM learning rate	Initial value 0.5, decaying by 0.949 each cycle
w_{max}, maximum allowed lateral synaptic weight	1.0 for excitatory, -1.0 for inhibitory
σ, neighbourhood spread	3.0
A_p, LTP rate	Various values used – see Chapter 8
A_m, LTD rate	-1.05 * A_p
τ_{ltp}, LTP time constant	10 ms
τ_{ltd}, LTD time constant	10 ms
Run time	
T_{int}, the integration time	9 ms
T_{out}, the time out parameter	30 ms

Table 1- Summary of neuron model, network and run parameters

The input layer is fully connected to the output layer with afferent (feedforward) connections. Afferent connection weights are set to a random value between 0.4 and 0.5

and there is a constant delay of 2.0 milliseconds on these connections. The output layer is recurrently connected and these lateral connections are sparse and follow a 'mexican hat' profile of short-range excitation and long-range inhibition for which there is experimental evidence in real cortex, and which has been used in many previous similar modelling studies (Kohonen, 1984, Miikkulainen et al., 2005). Excitatory and inhibitory connectivity is determined by probability functions based upon distance between the two neurons as given in equations (5) and (6)

$$p_{exc} = \exp(-\frac{dist}{sigma}) \quad (5)$$

$$p_{inh} = \exp(-\frac{sigma}{dist}) \quad (6)$$

Where:

p_{exc} is the excitatory connection probability (between 0 and 1.0)

p_{inh} is the inhibitory connection probability (between 0 and 1.0)

dist is the Euclidean distance between the neurons

sigma is the spread

For excitatory connections a sigma of 3.5 is used which gives a significant chance of connection at distances up to 5 units. At distances greater than this the probability is forced to zero. For inhibitory connections a sigma of 8.0 is used and at distances less than 4 units the probability is forced to zero. Figure 7 shows the profile of connection probabilities generated by this method.

Lateral connection weights are set to a random value between 0.3 and 0.4. The delays for the lateral connections are calculated according to the distance between the two neurons with added Gaussian noise (refer to Table 1 for details). Figure 8 shows a diagram of the network where afferent connections are shown from the input layer to one output neuron and lateral connections from one excitatory and one inhibitory neuron.

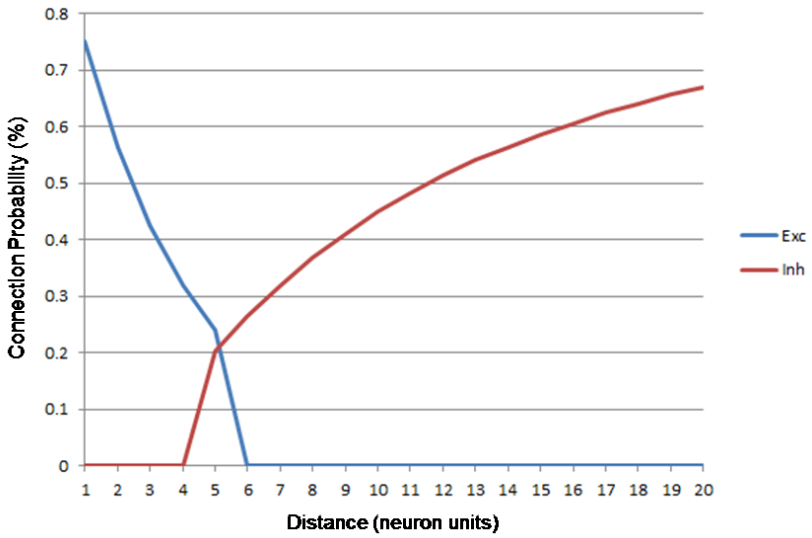


Figure 7- Connection probability profiles (16x16 output layer)

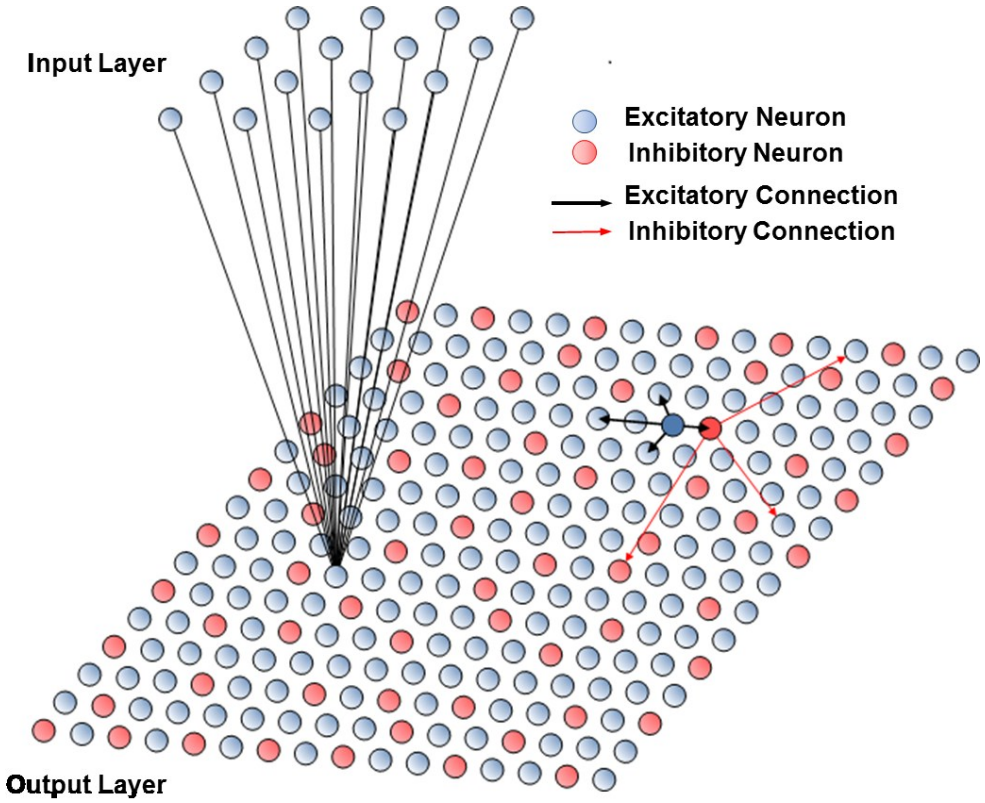


Figure 8- Motor Map Architecture

3.5 Motor Input Patterns

For the motor system exemplar training patterns representing 8 directions of movement (N, NE, E, SE, S, SW, W, NW) similar to those used in Marian (2002) were used. The patterns are encoded as 16 element vectors of spike times (one time for each input neuron) which are a mixture of 'salient' and 'noise' data. The rationale for the encoding

is based upon the methods of Maass (1997) where the spike timing relative to a reference time determines the contribution to the postsynaptic potential (PSP) of the output neuron. In Marian (2002), the convention was that the more recent the spike time (shorter latency), the greater the saliency and the larger the PSP contribution. This is adopted in the current work and the conversion of spike time to PSP is calculated using equation (7).

$$x_j = \exp(-\frac{(T_{int}-t_j-\tau_{da})}{\tau_{m,}}) \quad (7)$$

Where:

x_j is the contribution to the PSP from input neuron j.

T_{int} is the reference time (integration time)

t_j is the firing time of input neuron j

τ_{da} is a transmission delay on afferent (input-output) connection

$\tau_{m,}$ is the membrane time constant

The values used for the parameters T_{int}, $\tau_{da}$and τ_m are given in Table 1.

The parameter T_{int} is a computational convenience rather than having a particular biological counterpart. According to Maass' original formulation it is a constant that defines a small time window within which the relative timings of the input neurons *t* can be compared. Ruf and Schmitt (1998) also stated that T_{int} can be defined as a 'reference spike' provided by some other input to the neuron.

In the patterns for the current work noise is characterised by early firing of neurons between approximately 0 and 3 milliseconds. Salient information is represented by later firing of neurons between approximately 7 and 9 milliseconds (i.e. close to the integration time of 9 ms). An example pattern is shown in Figure 9: showing the burst of 'noise' data between 0 and 3 milliseconds and the 'salient' neurons firing closer to the

integration time T_{int}. Here the salient data makes the biggest contribution to the PSP. Under this scheme patterns that should be similar (for example directions N and NE) have been specifically created to encode some common salient information and patterns that are required to be dissimilar (for example directions N and S) do not encode any common salient information. To illustrate this, Figure 10 shows the encoding of example similar and dissimilar patterns. The patterns are composed of 12 noise values where neurons fire at early times with respect to the reference time T_{int} . The remaining values are salient and fire much closer to T_{int}. The patterns for N and S do not share any common salient information: the neurons which fire closer to T_{int} are completely different. However, the patterns for N and NE share 2 salient neurons: the firing times for neurons 12 and 14 are the same.

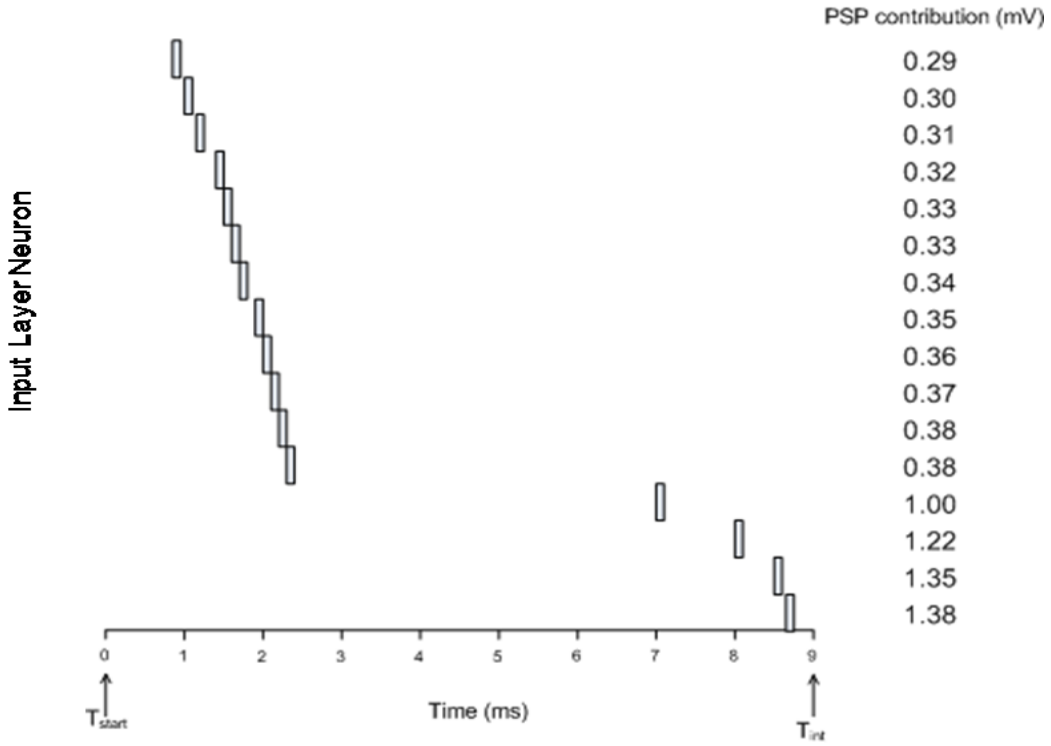


Figure 9- Latency coding of a 16 value input spike pattern

(adapted from Fig 4.7(a), Marian,2002)

Appendix A gives full details on how the exemplar motor patterns were created and

verified.

For initial prototyping, predefined datasets were used. A single training dataset consisted of 160 patterns: 20 instances of each of the 8 exemplar directional patterns, perturbed and randomly ordered. In the final system patterns are generated and presented randomly online. Full details of the process of creating the training datasets and the online generation process are given in Appendix A.

3.6 Learning

Of the previous implementations of spiking SOMs described in Chapter 2, only a few have used some form of Spike Timing Dependent Plasticity (STDP) which is the currently favoured model for learning in real neurons. In addition, not many works have allowed plasticity on both afferent and lateral connections nor on inhibitory connections. Following the success of the LISSOM architecture (Miikkulainen et al., 2005) in demonstrating the important role of plastic lateral connections and inhibitory plasticity in explaining many features of visual map development, the current work uses both plasticity on afferent and lateral connections and allows plasticity on the inhibitory connections as they are an essential to provide the dual aspects of cooperation and competition required for the map development.

Learning on the afferent connections is based upon the method originally created by Ruf and Schmitt (1998) and subsequently modified by Marian (2002). The form of the learning rule is given in equation (8).

$$\Delta w_{ij} = \eta \frac{T_{out} - t_j}{T_{out}} \left(\varepsilon_{ij} - w_{ij}\right) \cdot d(j, winner) \qquad (8)$$

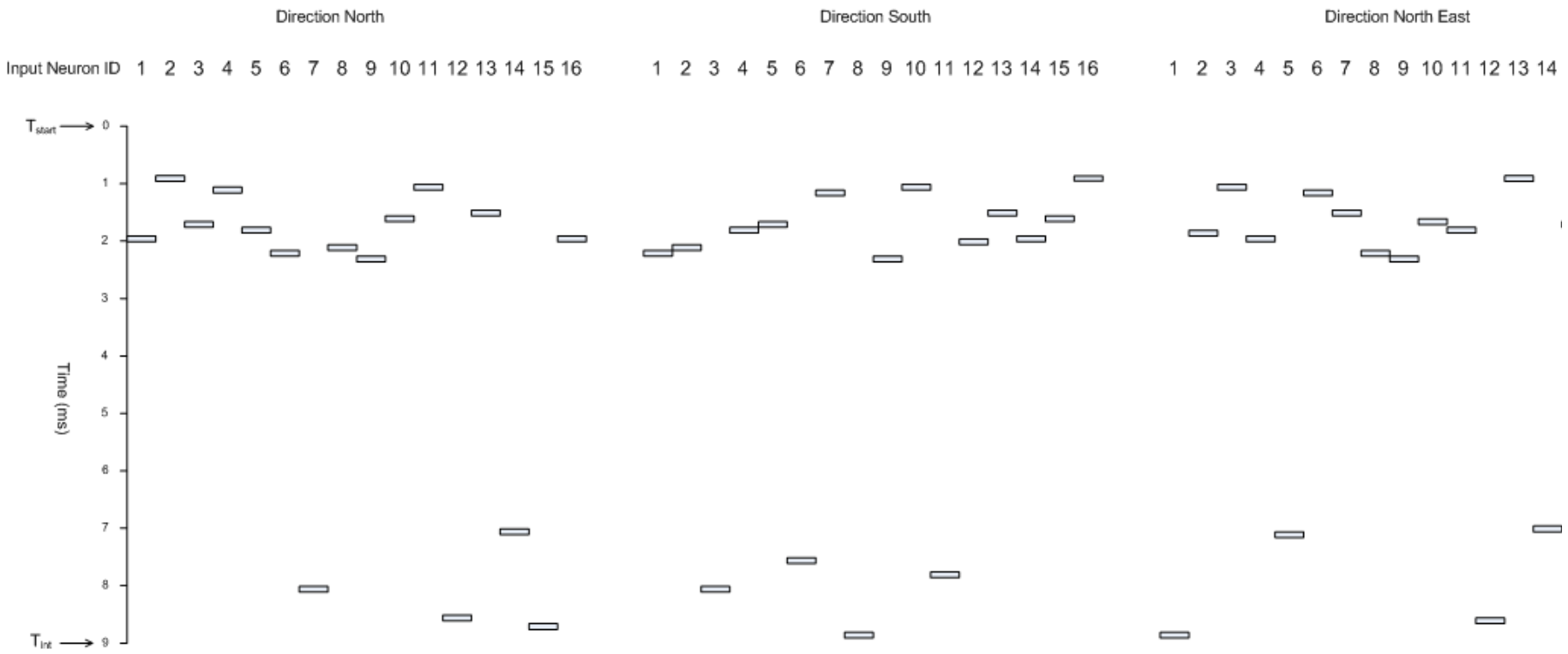


Figure 10- Comparison of similar and dissimilar input patterns

(adapted from Fig 6.3(a), Marian,2002)

Where:

w_{ij} is the weight on the connection between input neuron i and output neuron j.

η is the learning rate parameter

t_j is the firing time of output neuron j

T_{out} is a time out parameter

ε_i is the PSP contribution from neuron *i* calculated as $\exp(-\frac{T_{int}-t_i)}{\tau_m})$ where τ_m is the membrane time constant, T_{int} is the integration time and t_i is the firing time of the input neuron, *i*.

d is a neighbourhood function = $\exp(-(\frac{distance(j,winner)^2}{2\sigma^2}))$ where distance() is a function calculating the Euclidean distance between output neuron j and the winner neuron for the current learning cycle and σ is a neighbourhood spread parameter.

See Table 1 for details of the parameter values.

In the original Ruf and Schmitt (1998) implementation, the neighbourhood size was changed during training by manipulation of lateral connections and in Marian (2002) a traditional SOM neighbourhood with a reduction schedule was used. As the aim in the current work is to dispense with predefined reduction the learning rule is modified with a fixed, non-reducing spatial neighbourhood function implemented by a Gaussian function $d(j, winner)$ originally used by Pham et al. (2006).

In summary, the magnitude of the afferent weight update is determined by the following factors:

1. The value of ε_i which is determined by the presynaptic (input) neuron firing time
2. The difference between the value ε_i and the current weight w_{ij}
3. A temporal neighbourhood: synapses where the firing time of the postsynaptic

(output) neuron in response to the input pattern are quickest get a larger update

4. A spatial neighbourhood: synapses with firing postsynaptic (output) neurons located close to the current winner get a larger update

5. A learning rate parameter η which is reduced gradually during training

As discussed in the introduction to this chapter, equation (8) still employs a traditional learning rate parameter η which needs to be reduced gradually during training to control map development.

For learning on the lateral connections, Marian (2002) used an amended version of the Ruf and Schmitt (1998) afferent learning rule with slightly different formulations for excitatory and inhibitory connections. In their original work, Ruf and Schmitt (1998) did not use plasticity on lateral connections but manipulated the weights directly.

One problem with most kinds of Hebbian plasticity (including STDP) is that they are positive feedback systems. Correlated activity between a pre /post neuron pair strengthens the connection between them which makes further correlated activity much more likely. Without compensating adjustments, the weight increments will proceed unchecked. It is therefore usual to include some form of normalisation of all weights after a learning phase or apply hard limiting to a maximum / minimum value. For example, in Marian (2002) global normalisation was used to keep the lateral weights from getting too large. Neither global normalisation nor hard limiting is particularly biologically plausible (although there is some experimental evidence that forms of normalisation may occur in biological systems). Another problem specific to STDP is that in the general formulation (Song et al., 2000) the resulting weight distribution is bimodal which does not match with experimental results (van Rossum et al., 2000).

The lateral learning rule in the current work retains some elements of Ruf and Schmitt (1998) (e.g. the temporal neighbourhood) and has been modified to use STDP following

the basic method of Song et al. (2000) but incorporating weight dependent learning as suggested by van Rossum et al. (2000). Here the weight updates are dependent on the existing connection weight, with LTP updates being additive and LTD multiplicative. These rules results in a similar unimodal weight distribution to the experimentally observed one. In the original formulation of their rules van Rossum et al. (2000) also incorporate a weight dependent Gaussian noise term with a zero mean and standard deviation derived from experimental data. In the current work the noise term is omitted. The update rules for Long Term Potentiation (LTP) and Long Term Depression (LTD) are given as equations (9) and (10) respectively and are used for both excitatory and inhibitory connections.

$$\Delta w_{ij} = \eta \frac{T_{out} - t_i}{T_{out}} \left(w_{max} - w_{ij}\right) \cdot delta_{STDP} \cdot d(j, winner)\ ,\ w_{t+1} = w_t + \Delta w \qquad (9)$$

$$\Delta w_{ij} = 1 + \eta \frac{T_{out} - t_i}{T_{out}} \cdot delta_{STDP} \cdot d(j, winner),\ \ w_{t+1} = w_t * \Delta w \qquad (10)$$

Where:

w_{ij} is the weight on the connection between output neuron i and output neuron j

w_{max} is the maximum allowed lateral weight

η is the learning rate parameter

T_{out} is a time out parameter

$delta_{STDP}$ is the standard exponential STDP function:

$$A_p \cdot \exp\left(-\frac{t_j - t_i}{\tau_{ltp}}\right) \qquad \left(t_j - t_i\right) > 0$$

$$A_m \cdot \exp\left(\frac{t_j - t_i}{\tau_{ltd}}\right) \qquad \left(t_j - t_i\right) < 0$$

t_i, t_j are the firing times of presynaptic neuron i and postsynaptic neuron j respectively.

A_p, A_m are the STDP potentiation and depression rates respectively.

τ_{ltp}, τ_{ltd} are the time constants for potentiation and depression.

d is a neighbourhood function = $\exp(-(\frac{distance(j,winner)^2}{2\sigma^2})$ where distance() is a function calculating the Euclidean distance between output neuron j and the winner neuron and σ is a neighbourhood spread parameter.

See Table 1 for details of the parameter values.

These rules avoid the need for either global normalisation or hard limiting and ensure that the connection weights cannot change sign.

In summary, the value of the lateral weight update is determined by:

1. Presynaptic and postsynaptic neuron firing times (using the exponential STDP equation of Song et al., 2000)
2. The current weight (using update rules similar to van Rossum et al., 2000 but without noise)
3. A temporal neighbourhood: synapses where the firing time of the postsynaptic neuron in response to the input pattern are quickest get a larger update
4. A spatial neighbourhood: synapses with firing postsynaptic neurons located close to the current winner get a larger update
5. A learning rate parameter η which is reduced gradually during training

As for the afferent rule, equations (9) and (10) still employ a learning rate parameter η. The LTP rule given in (9) also incorporates a w_{max} parameter which is used to ensure that as the weights increase they asymptotically approach a maximum value. Later on in this thesis, Chapter 5 describes how an adaptive plasticity method was used to dispense with the learning rate for both afferent and lateral rules as well as w_{max}.

3.7 The Training process

The learning update cycle is based upon the methods used by Ruf and Schmitt (1998) and Marian (2002) and involves the following steps:

1. A direction pattern of 16 spike times is presented to input layer of the network and causes the input layer neurons to fire over an interval of 0ms up to T_{int} (the integration time)
2. From T_{int} up to T_{out} (the time out) the activity is allowed to propagate through afferent and lateral connections
3. After T_{out}, a 'winner' neuron is selected randomly from the group of output neurons that fired the quickest during the activity propagation period
4. Afferent learning is applied using equation (8)
5. Lateral learning is applied using equations (9) and (10)
6. The network is reset for the next pattern
7. The learning rate η for both afferent and lateral connections is decreased

The concept of the winner neuron is borrowed from a standard SOM network, except that in the networks here the selection is based upon time, not activation. Although there is no specific biological analogue, the idea behind this is that neurons which respond fastest to an input are gradually recruited to only respond to that pattern. As there will likely be several neurons which respond at the same time, the selection of a single winner is made randomly which allows non-selected neurons to become active for another pattern.

3.8 Scaling Up

The methods described previously in this chapter essentially replicate those of Marian (2002) with some amendments to the learning rules. The work described in this thesis is aimed at future implementation on robotic systems, particularly using the SpiNNaker neuromorphic hardware. As such, larger networks can be simulated and are likely to be required for autonomous robotics applications. Therefore it was important to

demonstrate that the motor map development system can work for a larger network. The scaling up amendments described in this section were actually done after the creation and testing of the visual map system (see Chapter 4) and prior to developing the map coupling (Chapter 5). However, the work is included in this chapter in order to keep all work on the motor system together. The amendments made are summarised as follows;

- Network size – the output layer was increased to 48x48
- Connectivity functions - for excitatory connections the amendment is straightforward. A slightly larger sigma value of 4.0 is used to ensure a higher probability of connections out to about a distance of 12 units. Further out than this the probability is clipped to zero. Due to the much larger size of network a more complicated method of assigning inhibitory connection probabilities was used in order to ensure a balance between enough inhibitory activity to ensure competition but also fairly sparse connectivity for performance reasons. This is summarised in Table 2

Distance (neuron units)	**Probability (%)**
12 units < distance < 15 units	exp(-*sigma*/*dist*) with *sigma* = 15.0
15 units =< distance < 30 units	0.7
30 units =< distance < 40 units	0.5
distance > = 40 units	0.2

Table 2 – Assignment of inhibitory connection probabilities for the 48x48 output layer

Figure 11 shows the new profile of connection probabilities generated by this method.

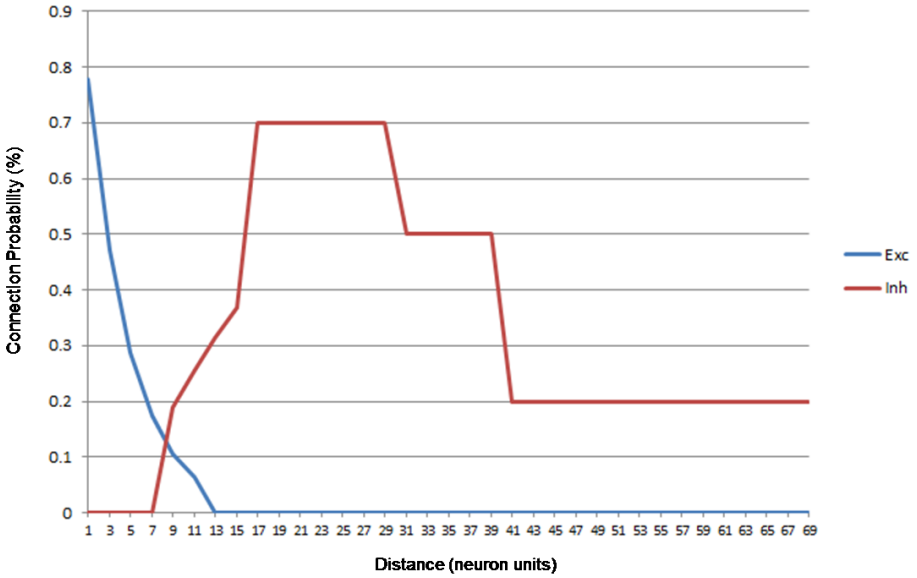


Figure 11– Connection probability profiles (48x48 output layer)

3.9 Conclusions

A full description of the experiments and detailed results for the motor map are described in Chapter 9, section 9.1 and a discussion of the results is given in Chapter 11. A summary of the results highlights is given below.

- The training process results in a directionally selective map after approximately 800 pattern presentations
- The afferent connections between input and output layers learn the characteristics of the input pattern vectors
- The majority of neurons in the output layer respond preferentially to one of the input patterns, although, as in real cortical maps there is some overlap
- The output layer response to each pattern is distinct in terms of spatial (neuron) and temporal (firing time)
- The output map is topographic – clusters of spatially located neurons respond to the same pattern
- Qualitatively similar results are achieved with both the 16x16 and 48x48 neuron output layers.

4 A Visual Cortical feature Map

4.1 Introduction

This chapter describes the features of the spiking neuron visual map architecture and SOM training process used to create a directionally selective visual map. The architecture of the visual map is similar to that described in Chapter 3 for the motor map, but with amendments in order to use different input data which is provided by the DVS 128 silicon retina capturing moving objects. A different learning method is also used specifically for developing directional selectivity from moving visual patterns and has been inspired by the work of Shon et al. (2004) and Wenisch et al. (2005). Similarly to the motor map the aim is to create a cortical-like map which can self-organise to form a topological map. Again, at this stage of the research the map development system does not have all of the desirable features mentioned in Chapters 1 and 2 with respect to adaptive learning. Later improvements and additions for adaptive learning are left until Chapter 5.

4.2 The Visual Map Architecture

The visual map network is similar to the SOM setup for the motor map but with an extra input layer that is required for the processing of visual data from the DVS 128 camera (see Section 4.4 for more details of the input data). Figure 12 shows a diagram of the network architecture. The first layer consists of 128x128 neurons (referred to as the 'Input' layer) and its purpose is merely to accept input spike data into the network without processing. The second layer consists of 64x64 neurons (referred to as the 'Retinal' layer) and its purpose is to achieve a down-sampling of the raw input data by half. The Input layer is connected to the Retinal layer with excitatory connections with fixed weights of value 1. These connections are such that a 2x2 Receptive Field (RF) from the Input layer is connected topologically to 1 neuron in the Retinal layer. These RFs are not overlapping, thus each neuron in the Retinal layer averages the activity

from 4 pixels in the Input layer. The box marked 1 in Figure 12 shows an example of one such set of connections. The neuron time constant and refractory period for the Retinal layer neurons are set to ensure that there is no multiple firing in the Retinal layer: i.e. any activity in the 2x2 group of input neurons results in 1 spike in the Retinal Layer neuron (see Section 4.3 for more details).

The cortical output layer consists of 116x116 neurons of which 20% are randomly assigned as inhibitory and 80% as excitatory. In contrast to the motor map setup, the Retinal layer is not fully connected to the output layer, but, in keeping with the approach of previous works modelling the visual system (e.g. LISSOM) each cortical neuron only 'sees' a patch of the Retinal layer called the Receptive Field (RF). In the current work a 7x7 square is used and so the RFs from each cortical neuron overlap: see the box marked 2 in Figure 12 for an example.

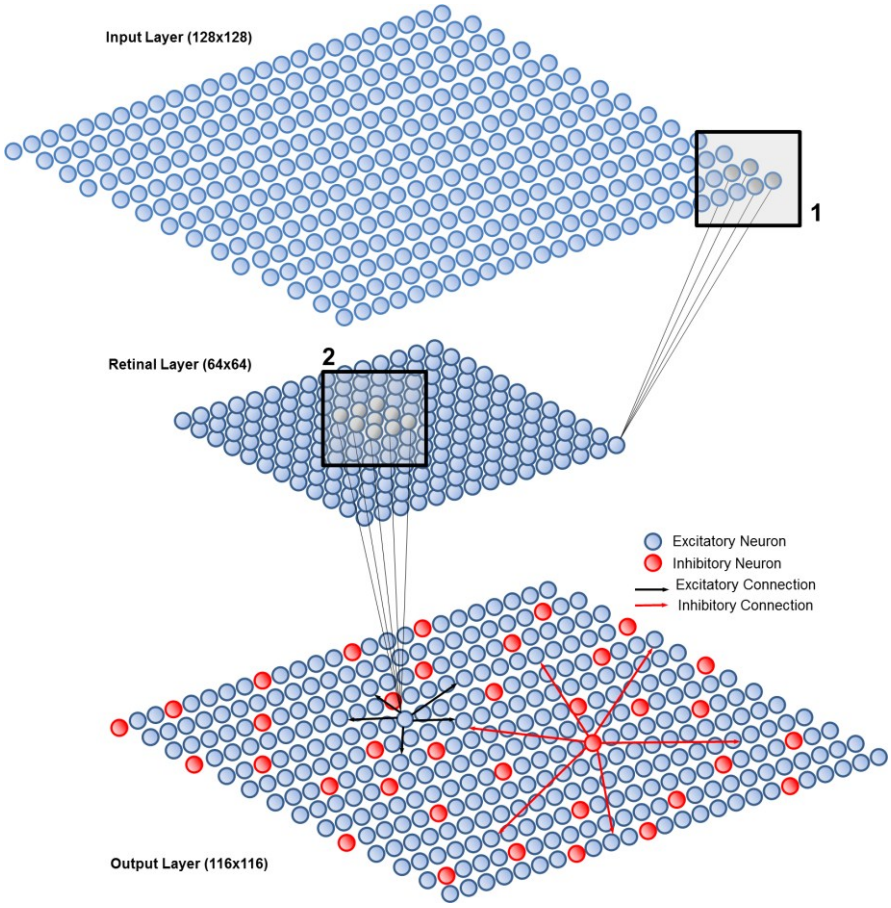


Figure 12- The Visual Map Architecture

There is a cortical magnification (ratio of cortical to retinal neurons) of 2. Afferent connection weights are set to an initial random value between 0.4 and 0.5. The output layer is recurrently connected: there are sparse lateral connections and these follow a 'mexican hat' profile of short-range excitation and long-range inhibition. Excitatory and inhibitory connections are determined by a probability function based upon distance between the two neurons as shown in equations (11) and (12).

$$p_{exc} = \exp(-\frac{dist}{sigma}) \quad (11)$$

$$p_{inh} = \exp(-\frac{sigma}{dist}) \quad (12)$$

Where:

p_{exc} is the excitatory connection probability (between 0 and 1.0)

p_{inh} is the inhibitory connection probability (between 0 and 1.0)

dist is the Euclidean distance between the neurons

sigma is the spread

For excitatory connections a sigma of 3.5 is used which gives a significant chance of connection at distances up to 5 units. At distances greater than this the probability is forced to zero. For inhibitory connections a sigma of 8.0 is used and at distances less than 5 units and greater than 21 units the probability is forced to zero. Figure 13 shows the profile of connection probabilities generated by this method. The profile for inhibitory connections used here is quite different to that used for the 48x48 motor map described in Chapter 3, Section 3.7 and the reason is because in the case of the motor map the input and output layers are fully connected therefore in order to achieve the competition required to make the topographic map the inhibitory influence needs to extend further. In the case of the visual map, the cortical layer is only connected to a subset of the Retinal layer, and thus the map that is form is more distributed in nature (for example, refer to Figs 4 and 5 in Chapter 2), therefore the inhibitory influence only

needs to extend a short way out.

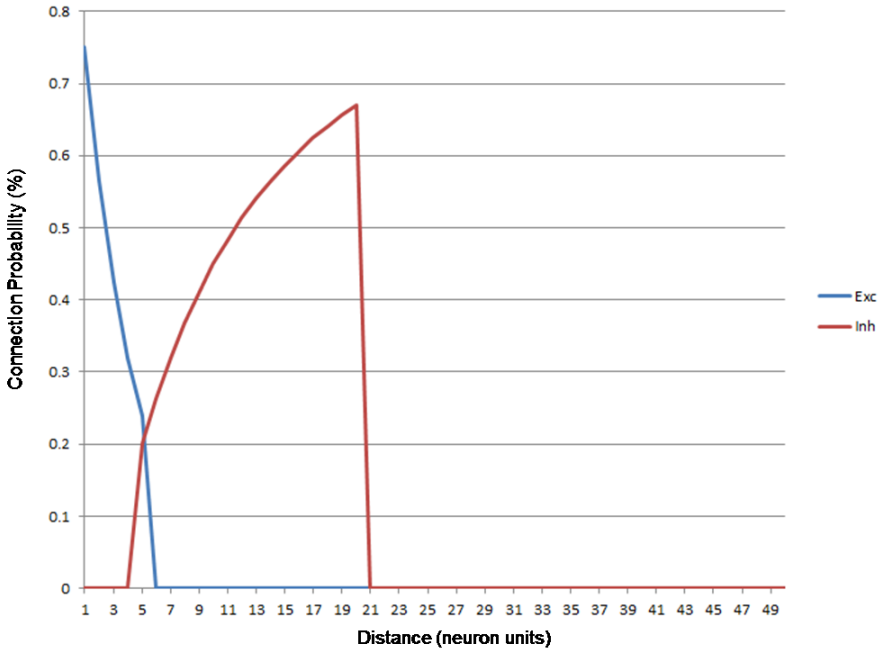


Figure 13- Connection probability profiles for the output layer

Lateral connection weights are set to an initial random value between 0.3 and 0.4.

Lateral connections incorporate delays which are calculated according to the distance between the two neurons with added Gaussian noise (refer to Table 3 for details).

4.3 The Neuron Models

A simple Leaky Integrate and Fire (LIF) model is used for the Retinal layer neurons to process the raw spikes from the Input layer and is given as equation (13)

$$\tau_{rm} \frac{dV_r}{dt} = -V_r \qquad (13)$$

Where:

V_r is the membrane voltage

τ_{rm} is the retinal membrane time constant

This is just a simple decaying voltage with spikes injected from connected neurons in the Input layer. When a presynaptic (Input) neuron fires the membrane voltage, V_r of the postsynaptic (Retinal) target neuron is increased as shown in equation (14).

$$V_{r\_new} = V_{r\_old} * w \quad (14)$$

Where:

V_{r_old} is the original membrane voltage

V_{r_new} is the updated membrane voltage

w is the synaptic weight

Synaptic weights are fixed at 1 for all connections. The membrane time constant is set at 10ms and the refractory period for retinal neurons is also 10ms. This setup ensures that the first firing of any Input neuron in the 2x2 group connected to the Retinal neuron will cause the retinal neuron to fire but immediate firing of other Input neurons in the group within the refractory period will not cause additional spikes in the Retinal neuron.

For the visual cortical neurons, the same basic Leaky Integrate and Fire (LIF) neuron model that was previously used for the motor model is used, but with one amendment. The model is described by equation (15).

$$\tau_m \frac{dV}{dt} = (g_e + g_i - V + N) \quad (15)$$

Where:

V is the membrane voltage

g_e is the contribution from excitatory synapses

g_i is the contribution from inhibitory synapses

N is exponential noise

τ_m is the membrane time constant

Following the method used in Shon et al. (2004) the noise term N is added to the model in the form of an approximation to the positive half of a zero-mean Gaussian as represented by equation (16).

$$\tau_n \frac{dN(t)}{dt} = -N(t) + 0.35[\mu_n + \sigma_n \left(\sqrt{\frac{1}{\tau_n}}\right) \eta(t)] \quad (16)$$

Where:

N is the noise at time t

μ_n is the mean of the noise

σ_n is the standard deviation of the noise

τ_n is the time constant

$\eta(t)$ is Gaussian white noise.

Synaptic dynamics are represented by equation (17).

$$\tau_s \frac{dg}{dt} = -g \tag{17}$$

Where:

g is the effective conductance for an excitatory or inhibitory synapse

τ_s is the synaptic time constant

When a presynaptic neuron fires the effective conductance (g) for excitatory and inhibitory synapses is updated as shown in equation (18).

$$g_{new} = g_{old} * w \tag{18}$$

Where:

g_{old} is the original effective synaptic conductance

g_{new} is the updated effective synaptic conductance

w is the synaptic weight

Table 3 gives a description of the neuron model parameters and their initialised values.

Parameter	Value
Neuron Model	
V_{reset}, reset voltage (retinal and cortical)	0 mV
$V_{ThreshR,}$ neuron threshold (retinal)	0 mV
$V_{ThreshC,}$ neuron threshold (cortical)	Randomly initialised as 1.0 mV plus noise normally distributed between 0 and 0.3 mV
τ_{rm}, membrane time constant (retinal)	10 ms
τ_m, membrane time constant (cortical)	5 ms
τ_s, excitatory/inhibitory synaptic time constant	5 ms
τ_n, noise time constant	5 ms
μ_n, noise mean	0.7
σ_n , noise standard deviation	0.5
τ_{dl} , delay on lateral synapses	Set as distance between pre and postsynaptic neuron plus noise added by a Gaussian with mean 0 and standard deviation 0.5
$\tau_{rrefrac}$, neuron refractory period (retinal)	10 ms
τ_{refrac}, neuron refractory period (cortical)	5 ms
Network Architecture	
N_{in}, number of neurons in Input layer	16384 (128x128)
N_r, number of neurons in Retinal layer	4096 (64x64)
N_{out}, number of neurons in output layer	13456 (116x116)
w_{aff}, afferent synaptic weights	Randomly initialised between 0.4 and 0.5
w_{lat}, lateral synaptic weights	Randomly initialised between 0.3 and 0.4 (exc) and -0.3 and -0.4 (inh)
Exc_p_{conn}, connection probability for lateral excitatory connections	Calculated as *exp(-dist/sigma)* where dist is the Euclidean distance between the neurons and sigma is 3.5
Inh_p_{conn}, connection probability for lateral inhibitory connections	Calculated as *exp(-sigma/dist)* where dist is the Euclidean distance between the neurons and sigma is 8.0
Learning Rules	
w_{max}, maximum allowed lateral synaptic weight	1.0 for excitatory, -1.0 for inhibitory (used in experiment 1 only)
A_{pa}, Afferent LTP rate	Various values used (see Chapter 8)
A_{ma}, Afferent LTD rate	-1.05 * A_{pa}
A_{pl}, Lateral LTP rate	Various values used (see Chapter 8)
A_{ml}, Lateral LTD rate	-1.05 * A_{pl}
τ_{ltp}, LTP time constant	11 ms
τ_{ltd}, LTD time constant	20 ms

Table 3- Summary of neuron model, network and run parameters

4.4 Visual Input Patterns

In the main, previous works have used artificially generated moving bar input to create directionally selective feature maps (for example, Bednar et al., 2003; Wenisch et al., 2005). A novel feature of the visual system in the current work is that the input is provided by real data from a DVS 128 silicon retina camera (Delbrück, 2008; iniLabs website, 2010). See Figure 14. This device has been developed within the domain of neuromorphic engineering and has only very recently begun to be used in specific biologically-inspired machine vision applications. For example, the work of Davies et al (2010) which developed a line-following robot using the DVS camera and a prototype 4-chip SpiNNaker neuromorphic board and also the more recent work of Galluppi et al. (2012) which used the camera as an input device to an orientation selective visual system also implemented on SpiNNaker. The benefits of such a device in robotics applications are manifold. Firstly, the input is frame-free: it consists of individual packets which hold an address (encoding the spatial position) and a timestamp for a spike event. Therefore it is not necessary to process whole image scenes at a time, only events. Secondly, for our purposes, minimal pre-processing is required as the input directly encodes spike events and so they are in a form which can be directly applied to the visual network. Thirdly, an event is only generated when something changes and so no time or resources are wasted processing visual information when nothing has actually happened. Lastly the DVS 128 output is illumination independent as the triggering of a spike event is based purely upon pixel-level changes in the input. This is an extremely important issue for vision systems in autonomous robotics as they need to be robust to changing light levels. The camera outputs raw events in AER (Address-Event Representation) format which consist of a 4-byte address and a 4-byte timestamp. The address encodes a spatial x, y position (in the range [0,127] [0,127] for the event and also an event polarity of 1 or -1 signifying ON or OFF events respectively.

Therefore the camera can register both when a pixel is activated and deactivated.

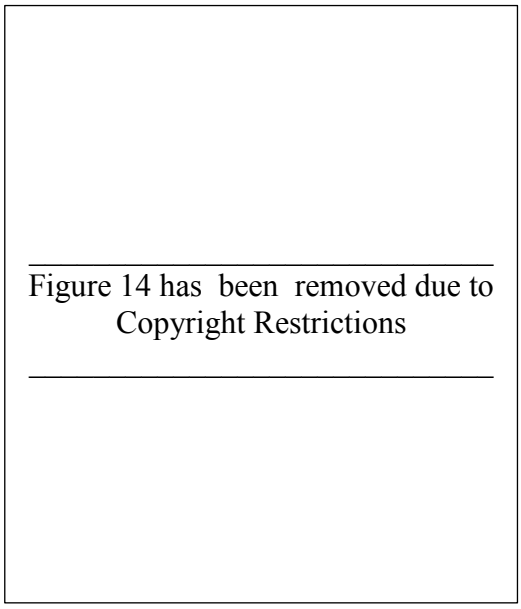


Figure 14- The DVS128 Camera
(iniLabs Website, 2010)

In the current work the bulk of the experimentation was done with logged data from the camera: pre-recorded sequences of objects moving in one of 8 directions (N, NE, E, SE, S, SW, W, and NW). The recordings were made using the jAERViewer interface which allows live visualisation, playback and logging of AER data from the camera (jAER SourgeForge wiki, 2012). The data are saved as version 2.0 AER files with an .aedat extension and consist of a text header and event data written in binary format as int32 address, int32 timestamp (8 bytes total). The timestamps are recorded with a 1 microsecond resolution. Figure 15 shows an example screen capture from a recorded sequence of an object moving in the North direction. ON events are white/light grey and OFF events black/dark grey. The medium grey pixels are where there has been no change from the previous timestep.

In order to convert the DVS 128 AER events into spikes in the Input layer of the visual system, some (currently experimental) features of the Brian simulator have been used.

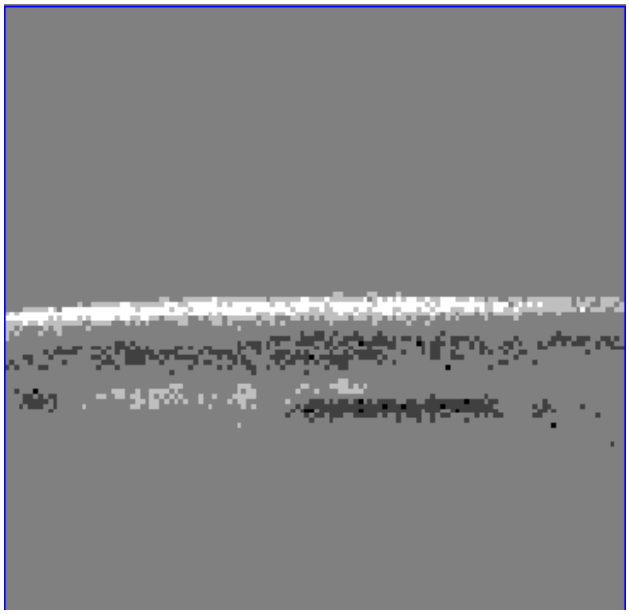


Figure 15-Example DVS 128 recorded AER data

The Brian module *experimental/neuromorphic/AER.py* has functions for reading the contents of an entire .aedat file and converting the events to Python lists of integer addresses and timestamps (load_aer()) and also for extracting the x, y coordinates and polarity of an event from its address (extract_DVS_event()). These features are used in the current work to load and process an .aedat file into lists of x coordinate, y coordinate, polarity and timestamp. The DVS camera produces a very large number of spike events, even for a simple object such as that shown in Figure 15– approximately 70,000 spikes over a duration of about 1.5 seconds. Handling this quantity of spikes is less of a problem for purely software implementations (although there are performance hits in run time), however for future deployment onto neuromorphic hardware there are limitations to consider as the hardware has an upper limit to the number of neurons that can be modelled and limitations on the bandwidth for handling input data. This issue was raised in Davies et al. (2010). Their solution to this problem was space subsampling on the raw 'image' from 128x128 down to 16x16 for both the ON and OFF events. The recent work of Galluppi et al. (2012) has used an FPGA board as an intermediary to sample the DVS data prior to injecting spikes into the SpiNNaker network.

The method employed in the current work is also a form of space subsampling but implemented as part of the neural network (as described in sections 4.2 and 4.3). Also it was decided to only use ON events as the OFF events would contribute no extra information in the current setup. Once the ON spike events have been passed through the Input layer and down-sampled in the Retinal layer to a resolution of 64x64, the number of spikes has been reduced to approximately 20% of the original. Figure 16 shows comparison raster plots of raw spikes in the Input layer (Figure 16a) and spikes generated in the Retinal layer (Figure 16b) for the North pattern sequence. It can be seen here that the down-sampling does not affect the character of the information. Tests have also been done sampling down even further to 32x32 and the characteristics of the input are still preserved. Note that in visual training there is no predefined sequence of patterns and instead patterns are selected randomly from the 8 exemplar sequences.

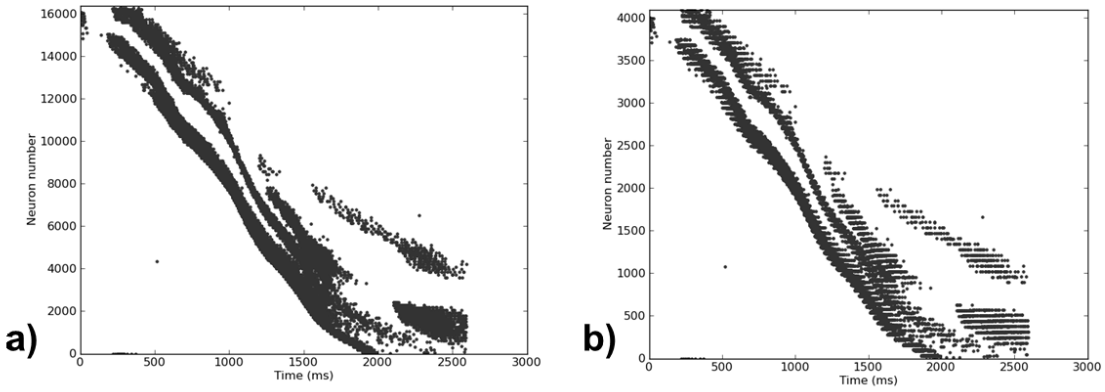


Figure 16- Comparison of a) 128x128 and b) 64x64 spike events for North sequence

Although the experiments with the visual system were performed using logged DVS 128 data, it was important to establish potential methods for dynamic input live from the DVS camera and so during the course of the research a system was prototyped for this. As this work is not directly used in the experiments, but is likely to be of interest to readers it is presented separately in Appendix B.

4.5 Learning

The review of previous works in Chapter 2, Section 2.4 highlighted the fact that, even to date there is debate as to what features are required to develop directional selectivity. Many works have achieved DS maps using various combinations of afferent (feedforward) and lateral learning either with or without LGN ON/OFF mechanisms. In the current work both afferent and lateral learning are used, and the methods are inspired by the work of Shon et al. (2004) and Wenisch et al. (2005) which both proposed an STDP learning regime with an asymmetric time window. Wenisch et al. (2005) showed how robust cortical directional selectivity could arise using such a learning regime combined with distance dependent delays between cortical neurons and their methods are used in the current work as they fit well with the cortical architecture that has been used. that has been used. The STDP functions for the LTP and LTD elements are given as equations (19) and (20).

$$y = exp\left(-\frac{\Delta t}{\tau_{ltp}}\right) \tag{19}$$

$$y = -\frac{\Delta t}{\tau_{ltd}} exp\left(1 - \frac{\Delta t}{\tau_{ltd}}\right) \tag{20}$$

Where:

y is the magnitude of the weight change

Δt is (firing time of the presynaptic neuron –firing time of the postsynaptic neuron)

τ_{ltp} is the LTP time constant

τ_{ltd} is the LTD time constant

See Table 3 for details of the parameter values.

These equations result in an asymmetric STDP window where the LTP part is an exponential curve as usual (equation 19) , but the LTD part is represented by an alpha function (equation 20).

Figure 17 shows a plot of the functions. This means that the LTD part of learning is deeper and persists for longer than LTP, and according to Wenisch et al. (2005) is an important mechanism for causing directional selectivity. The rationale for this is explained in Figure 18. Figure 18 shows a scenario where activity is moving left to right across a row of cortical neurons. The vertical red arrows indicate inputs from connections which get strengthened as they contribute to a spike in the central postsynaptic neuron within the LTP time window (10ms). The black arrows in the lower half of the figure show the effects of synaptic delays: as the wave of activity gets closer to the postsynaptic neuron, there is a shorter delay and a stronger contribution to the Postsynaptic Potential. Once the activity has passed over the postsynaptic neuron the stronger, longer LTD comes in to play as later inputs spike after the postsynaptic neuron and are thus weakened (vertical blue arrows).

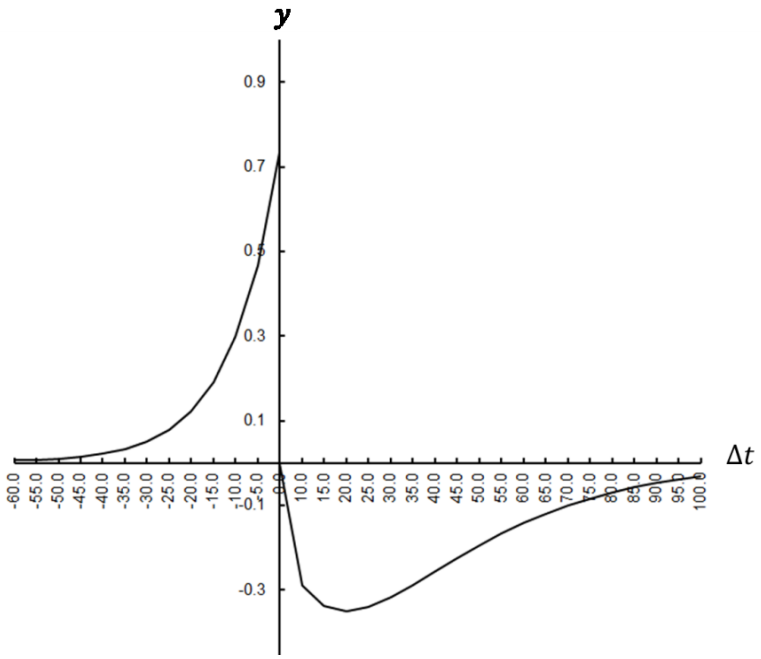


Figure 17- The Asymmetric STDP time window

Figure 18 has been removed due to Copyright Restrictions

Figure 18- How the asymmetric time window causes directional selectivity
(reproduced from Fig 4. In Wenisch et al., 2005)

If the same directional input is repeated over time, the weights of inputs coming from the direction of the moving wave (the 'preferred' direction) will get strengthened and those on the opposite side will decrease even further, resulting in a directionally selective response. Activity coming in from the opposite direction (the 'null' direction) is less likely to cause a spike in the postsynaptic neuron as the weights on that side are weakened.

The weight update rules for Long Term Potentiation (LTP) and Long Term Depression (LTD) used in the current work are given as equations (21) and (22) respectively and are used for both afferent excitatory and lateral excitatory and inhibitory connections.

$$\Delta w_{ij} = \eta\left(\left(w_{max} - w_{ij}\right) \cdot A_p\right),\ w_{t+1} = w_t + \Delta w \quad (21)$$

$$\Delta w_{ij} = 1 + \eta(A_m),\ \ w_{t+1} = w_t * \Delta w \quad (22)$$

Where:

w_{ij} is the weight on the connection between presynaptic neuron i and postsynaptic neuron j

w_{max} is the maximum allowed weight

A_p and A_m are the LTP and LTD learning rates

η is a learning rate parameter

These rules use the same weight dependent update methods (due to van Rossum et al, 2000) as used for the motor map learning and avoid the need for either global normalisation or hard limiting and ensure that the connection weights cannot change sign.

In summary, the value of the weight updates is determined by:

1. Presynaptic and postsynaptic neuron firing times (using the asymmetric STDP equations of Wenisch et al., 2005)
2. The current weight (using update rules similar to van Rossum et al., 2000 but without noise)
3. A learning rate parameter η which is reduced gradually during training

Equations (21) and (22) still employ a learning rate parameter η, and the LTP rule given in equation (21) also incorporates a w_{max} parameter which is used to ensure that as the weights increase they asymptotically approach a maximum value. Chapter 5 describes how an adaptive plasticity method was used to dispense with both of these.

4.6 The Training process

The learning update cycle involves the following steps:

1. A direction is randomly selected
2. The corresponding .aedat file is loaded and the AER events processed to lists of x,y coordinates and timestamps
3. ON events are selected out and applied to the Input layer
4. Neural processing begins
5. Spikes are processed through the Retinal layer and propagate into Output

layer

6. Afferent and Lateral learning is applied continuously using equations (19) and (20)
7. Once all spikes in the sequence have been processed, the network is reset for the next pattern

In keeping with the dynamic nature of the input, the training sequence is quite different to that used for the Motor map. Instead of separate pattern presentation and learning segments, STDP learning is applied during the entire time the spikes are moving across the network.

4.7 Conclusions

A full description of the experiments and detailed results for the visual map are described in Chapter 9, section 9.2 and a discussion of the results is given in Chapter 11. A summary of the main features of the results is given below.

- The method of development of directional selectivity is validated by demonstrating a difference in response after training to preferred and null directions
- A full map representing 8 directions of motion is formed
- The changes in both the afferent and lateral weights are directionally selective
- It was found that a visual map is present even before training. The initial map has responses that are similar to maps produced from experimental results, but with a lot of overlap between responses to different patterns. Repeated training with random presentations from the set of exemplars refines and reorganises the response so that it is sparser (action of the asymmetric rule plus competition mediated via lateral inhibition) and more distinct. This fits well with experimental findings that rudimentary maps are present at birth / eye-opening and are subsequently refined by experience.

5 Adaptive Plasticity

5.1 Introduction

A central aim of this research was to introduce amendments to traditional Self-Organising Feature Map (SOFM) methods in order to make them more useful for training sensorimotor controllers for autonomous robots. Chapter 1 outlined three main drawbacks of the commonly used SOM method: control of training by direct manipulation of learning rate and neighbourhood parameters; the use of predefined datasets; the general lack of adaptivity in the trained map. An important requirement for use in a robotics application (particularly a bio-inspired one) is that the process needs to be self-regulating as we want the robot to learn autonomously from the information available in its environment, and, as far as possible not influenced by human decisions. Chapters 3 (motor system) and 4 (visual system) described the basic SOFM methodology (based upon existing techniques but adapted for use with spiking neurons) which used some elements of traditional SOM training. The current chapter describes work that was done to develop a method of 'adaptive plasticity' which has been used primarily to improve the autonomy of map training. Subsection 5.2 describes how a 'plasticity resource' (PR) is modelled as a global parameter which expresses the rate of map development and is related directly to learning on the afferent (input) connections. This is used to control map training in both the motor and visual maps in place of a traditional learning rate parameter. Subsection 5.3 addresses the point raised in Chapter 1 regarding the general lack of structural plasticity (synaptic creation and pruning) in previous SOFM works and proposes a training process incorporating both functional and structural plasticity. Section 5.4 concludes the chapter by summarising the main points and results.

5.2 The Plasticity Resource

Rationale

In the current work a bio-inspired replacement for traditional learning rate reduction schedules has been inspired by considering the type of process that might determine when a particular map refinement phase is 'finished' in a real cortical map. In a study of the formation of the retinotectal map in goldfish, Schmidt (1985) noted that there was a correlation between the rate of synaptogenesis and the amount of disruption to map organization: as time goes on there is less synapse refinement and thus less potential for disruption. Schmidt explained this by a change in arbor size (spread of dendritic projections) of the retinal ganglion cells. Reduction in arbor size over time means that there are effectively less potential connections available to be made. In very general terms regulation of plasticity in development can be viewed in terms of a 'resource' being consumed over time where the level of the resource influences how much plasticity is allowed. Van Ooyen (2001) reviewed the theory behind competitive synaptogenesis and various models that existed at the time. The majority of the reviewed models are based upon the concept of 'consumptive' competition, i.e. there is a limited resource (a Neurotrophic Factor or NTF) which is consumed when connections are made thus limiting the potential for further connections. Such processes involve both electrical and chemical regulation.

Modelling the Plasticity Resource

In the current work, a Plasticity Resource (PR) model was developed based upon some of the above concepts but initially considered only in relation to functional plasticity (weight changes). Subsection 5.3 discusses a methodology for structural plasticity (synapse creation and pruning) into which the PR was easily incorporated.

Network development is monitored and controlled by a 'Plasticity Resource' (PR) parameter which can be viewed as an abstraction of an NTF. The PR is based upon a

simple model where global network activity (i.e. the LTP/LTD processes stimulated by input patterns) directly controls the level of the plasticity resource which in turn is used to regulate weight changes over the course of training. The method used here is quite different to those developed by previous researchers which used various ways to measure the progress of learning based upon distances between input and weight vectors (Shah-Hosseini and Safabakhsh, 2000; 2001; Berglund and Sitte, 2006; Miyoshi, 2005; 2007; 2008; Berglund, 2010; Shah-Hosseini, 2011). An important feature of the current method is that as it is based upon the levels of LTP/LTD, it is easily applied to a spiking neural network and furthermore, does not require the training inputs to be in the form of pattern vectors. This will be demonstrated by its later application to visual system training where the input is in the form of spike events (see Chapter 9). The inspiration for using the levels of LTP/LTD to monitor training comes from the work of Harris et al. (1997) which implemented a consumptive scheme to specifically model Ocular Dominance map formation and included Hebbian LTP/LTD processes controlled by the consumption of an NTF-like resource. In the Harris model synaptic weights change as a consequence of Hebbian LTP and LTD processes. Each individual neuron has an allocation of NTF spread amongst its synapses. There is positive feedback between LTP on the afferent synapses and uptake (i.e. consumption) of the NTF thus causing competition between the synapses to use the NTF held by the neuron. The rate of LTP is also increased as the amount of NTF acquired increases. In the current work a simpler scheme is used whereby there is one global NTF pool or Plasticity Resource (PR). Also the PR can both increase and decrease by the direct action of LTP and LTD on the afferent connections.

In considering how to model this new Plasticity Resource it was hypothesised that the amount of weight change in response to input patterns should level off once a sufficient number of the input patterns had been presented. Early on in the map development

process the network activity is high (the map is learning novel features in the input). When the network has learned the range of patterns in the input, the activity should level off. To verify this for the motor map scenario an experiment was run for 10 training cycles (1600 patterns) using the methods described in Chapter 3, and data was collected on the LTP/LTD changes on all connections. Figure 19 shows the total cumulative weight updates per cycle for afferent connections only. The weight updates for lateral LTP and LTD showed a similar but much less distinct pattern which can be explained by the fact that it is the direct response to the input patterns which is the main driver for learning. The lateral activity is an indirect response to the activity coming in on the afferent connections.

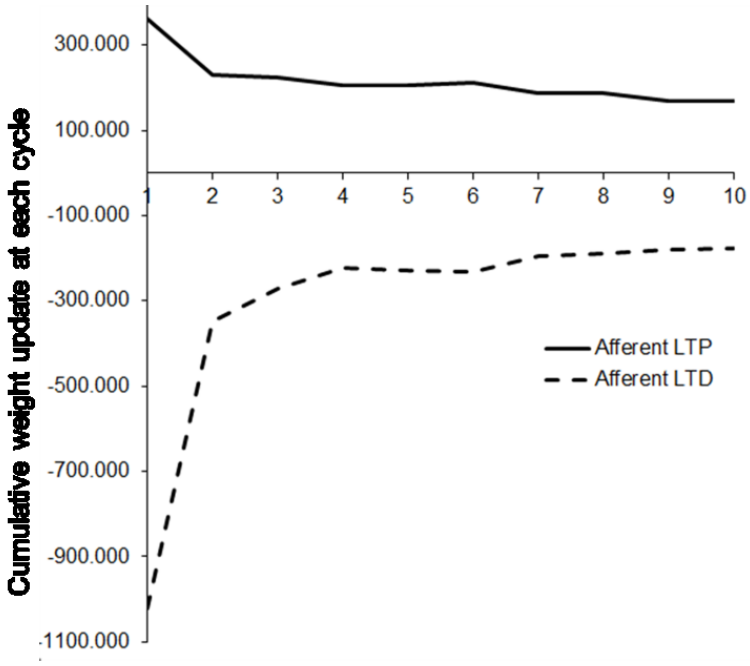


Figure 19- Relationship between LTP and LTD on Motor Map Afferent Connections

Figure 19 shows that at the beginning of training the amount of afferent LTP relative to LTD is quite different, but as training progresses this balances out and by cycle 10 they are more or less equal. To better express the stabilisation of afferent learning in terms of the relationship of LTP and LTD processes, Figure 20 shows the proportion of LTP to

LTD weight updates over the 10 training cycles which exhibits a distinct pattern of exponential increase and levelling off.

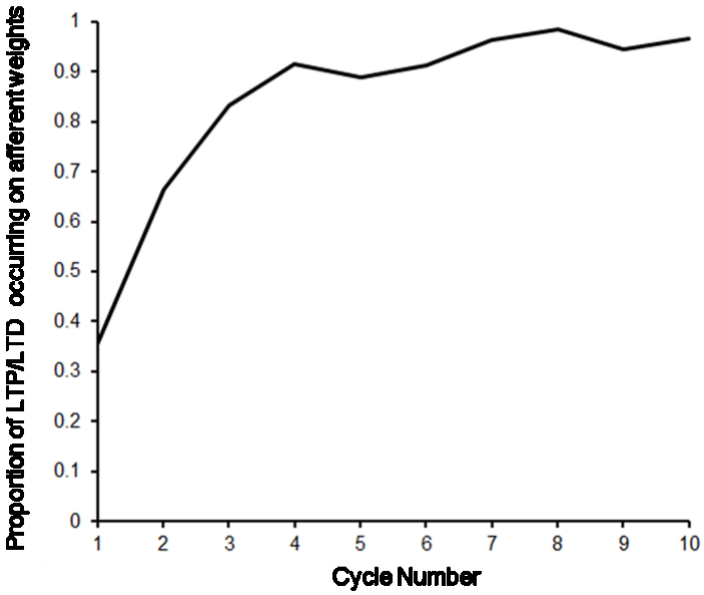


Figure 20- Ratio of LTP to LTD on Motor Map Afferent Connections

The ratio of LTP to LTD is calculated using equation (23).

$$ratio_{ltp\_ltd} = \frac{\min(abs(\sum LTP), abs(\sum LTD))}{\max(abs(\sum LTP), abs(\sum LTD))} \quad (23)$$

Where:

$ratio_{ltp\_ltd}$ is the calculated ratio

ΣLTP is the current sum of LTP weight changes

ΣLTD is the current sum of LTD weight changes

Absolute values of ΣLTP and ΣLTD are taken, with the minimum of the two as the numerator and the maximum as the denominator to ensure that the ratio is positive and less than 1.0. A similar process was performed to establish if the same behaviour occurred in visual map development. Using the methods described in Chapter 4. An experiment was run presenting 100 patterns and data was collected on the LTP/LTD changes on all connections. Figure 21 shows the proportion of LTP to LTD weight

updates over the 100 patterns. This also shows a pattern of exponential increase and levelling off although it is not as distinct as that shown in Figure 20. However, it was discovered that normalising the ratio so that the values lie between 0 and 1.0 gives similar behaviour to that shown in Figure 20.

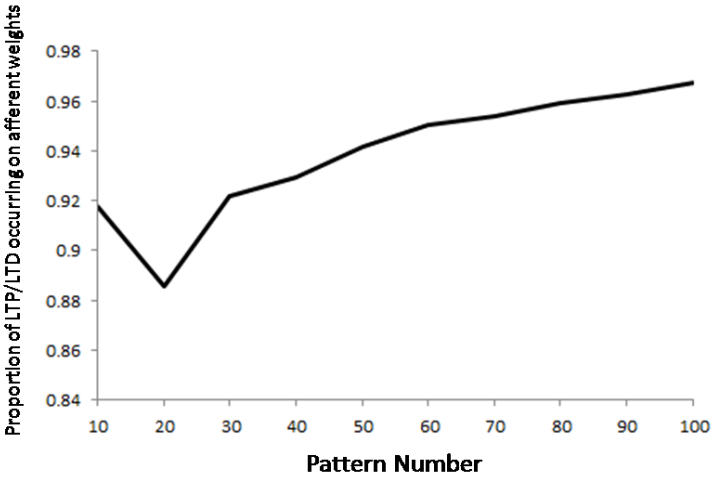


Figure 21- Ratio of LTP to LTD on Visual Map Afferent Connections

The normalisation is calculated using equation (24).

$$x_{norm} = \frac{(x - minx)}{(1.0 - minx)} \tag{24}$$

Where:

x is the value to be normalised

x_{norm} is the normalised value

minx is the minimum possible value of the ratio $ratio_{ltp\_ltd}$

The value of *minx* is updated by checking if the current value of $ratio_{ltp\_ltd}$ is less than the previous lowest value. In the early stages of training *minx* tends to fluctuate but settles into a stable minimum value after a few training iterations.

Equation (25) shows how the global 'plasticity resource' (PR) is calculated for both motor and visual training.

$$PR = 1.0 - norm(ratio_{ltp\_ltd}) \quad (25)$$

Where:

PR is the plasticity resource value

$ratio_{ltp\_ltd}$ is the LTP-LTD ratio as calculated by equation (23)

$norm$ is the normalisation applied as per equation (24)

The final PR value is taken as 1.0 minus the normalised $ratio_{ltp\_ltd}$ so that the PR value decreases over time and its action is to reduce the amount of learning. The PR value is updated using this calculation after presentation of each individual pattern once the weight updates have been completed.

Adjustments to Motor Map Learning Rules

In order to dispense with the learning rate parameter (η) used in both the afferent and lateral weight update rules and also the w_{max} parameter used to keep the lateral weight updates in check, new learning rules were created incorporating the PR value. These are given as equations (26) (afferent updates), (27) and (28) (lateral updates).

$$\Delta w_{ij} = PR \cdot \frac{T_{out} - t_j}{T_{out}} \left(\varepsilon_{ij} - w_{ij}\right) \cdot d(j, winner) \quad (26)$$

$$\Delta w_{ij} = PR \cdot \frac{T_{out} - t_i}{T_{out}} \cdot delta_{STDP} \cdot d(j, winner)\,,\ w_{t+1} = w_t + \Delta \mathrm{w} \quad (27)$$

$$\Delta w_{ij} = 1 + PR \cdot \frac{T_{out} - t_i}{T_{out}} \cdot delta_{STDP} \cdot d(j, winner),\ \ w_{t+1} = w_t * \Delta w \quad (28)$$

Where:

PR is the plasticity resource value as calculated by equation (25) and the other parameters are as described for the original learning rules in Chapter 3, Section 3.5.

Adjustments to Visual Map Learning Rules

In order to dispense with the learning rate parameter (η) used in both the afferent and lateral weight update rules and also the w_{max} parameter used to keep the lateral weight updates in check, the original weight update rules were modified to incorporate the PR value. These are given as equations (29) and (30) which are used for both afferent and lateral updates.

$$\Delta w_{ij} = PR \cdot A_{pre},\ w_{t+1} = w_t + \Delta w \quad (29)$$

$$\Delta w_{ij} = 1 + (PR * A_{post}),\ w_{t+1} = w_t * \Delta w \quad (30)$$

Where:

PR is the plasticity resource value as calculated by equation (25) and the other parameters are as described for the original learning rules in Chapter 4, Section 4.5.

5.3 Structural Plasticity

Rationale

Rewiring plasticity should be an important area of study in the development of self-organising maps, particularly in the case of autonomous robotic applications which are trying to mimic the features of real brain development and strike a balance between adaptivity and persistence. Few previous studies specifically include full rewiring. A notable example are the various works by Elliott and Shadbolt (1998a; 1998b; 1999) which created models of activity-dependent synapse formation and pruning based upon competition for an NTF-like resource. A later work implemented their model in a controller for a Khepera robot and demonstrated that allowing activity-dependent rewiring enabled the robot to recover its initial obstacle avoidance behaviour when sensory input was partially disabled (Elliot and Shadbolt, 2001).

The LISSOM models of Miikkulainen and collaborators (for example, Miikkulainen et

al., 2005) included pruning (but not creation) of connections once they reached a threshold weight. The recent work of Bamford (2009) included a model of the development of ocular dominance maps and included full rewiring where both synaptogenesis and pruning were allowed. The main focus of Bamford's work was to develop a rewiring methodology for implementation in neuromorphic hardware, but part of the experimental work included a small software modelling study of ocular dominance formation demonstrating the benefits of rewiring in allowing the network to retain some previous learning if the input changed.

In the current work the aim is to use rewiring to both improve adaptivity and also network efficiency. It is a feature of real biological development that there is a peak phase of synaptogenesis where many connections are formed, followed by a phase of massive pruning leaving only connections which serve a useful purpose. Most self-organising map studies, even if they do incorporate pruning, do not include any results to show if the final network is representing the input in a more efficient way: i.e. that the pruning results in sparser connectivity compared to a network without it. Although there is no particular rationale why biological systems would be more efficient having pruned unused connections, in the case of autonomous robotic applications implemented in neuromorphic hardware it is a very important feature to be able to create self-organising maps with the capability to retain only useful connections and where possible, reduce the initial connectivity whilst still achieving the desired result.

The Rewiring Model

In the model developed for the current work, lateral connections can be created and pruned as well as undergo weight changes. Applying rewiring to afferent connections has not been considered here, although the methods would be equally applicable.

Each cortical neuron has the potential to synapse with many neighbouring neurons, however, all-to-all connectivity is not possible due to geometric constraints as suggested

by Chklovskii et al. (2004). This is based upon the notion that the dominant form of rewiring plasticity is the forming of connections between neurons that are already close rather than significant axonal growth. Initial connectivity is determined using the same methods described in Chapters 3 (motor system) and Chapter 4 (visual system). Probabilistic connectivity functions which take into account the distance between neurons are used resulting in sparse connectivity. For the formation of new connections later in the learning process, the Euclidean distance between neurons is used as the geometric constraint. In addition each neuron has the capacity to sustain only a fixed number of actual synapses with other neurons. This is similar to the implementations of Jun and Jin (2007) and Bamford (2009). In an early model of synaptogenesis and pruning, Changeux and Danchin (1976) introduced the concept of connection states which change during the course of network development. Similarly, in the current implementation, connections are considered as being in one of three states:

Potential – a connection is possible but has not been activated yet

Active – a connection undergoing plasticity by STDP

Regressed – a connection which was once active but which has been pruned due to insufficient activity.

During the course of learning, the allowed transitions between states that can occur are:

Potential → Active – The rationale for this is based upon Chklovskii et al. (2004) which discussed the scope for rewiring without involving large amounts of axonal or dendritic growth. They estimated that the potential connectivity of this type between neurons in a cortical column is practically all-to-all and that the connection between these close neurons could be made by extending a spine or synaptic bouton. In the current work this scenario is modelled by checking the connectivity between all neurons firing during a pattern presentation. If they are not already connected and have both fired within the STDP time window and they meet the geometric constraints used to set

up the initial connectivity (as described in Chapter 3, section 3.4 for the motor map and Chapter 4, section 4.2 for the visual map) and the presynaptic neuron has not reached the maximum number of connections allowed, then a new connection is formed.

New connections are created with a random weight and delay in exactly the same way as for initial network setup.

Active → Regressed – If there is uncorrelated pre and post synaptic activity the connection is weakened by STDP until it falls below a threshold and is pruned.

Regressed → Active – If the input activity changes regressed connections can be reactivated. Again this is subject to the maximum connection constraint. Although it is considered as a separate state, in fact a regressed connection is treated no differently to a potential connection.

All Active connections undergo STDP as normal.

Table 4 gives a summary of the parameters and initial values for the rewiring model in the case of the prototype motor system (16x16 cortical map layer) and visual system (116x116 cortical map layer). An important point to note about the current model is that rewiring is completely activity dependent and is controlled only via the action of STDP. Synapse creation and destruction are not probabilistic and there is no specific schedule for rewiring, as used, for example by Bamford (2009). Instead, only the dynamics of the weight changes influence the creation and destruction of synapses: the action of correlated pre and postsynaptic firing can create a new synapse but unless the correlated activity is sustained the connection will be pruned. Likewise, connections made at initialisation that do not exhibit correlated activity will be discarded. The precise interaction between weight changes and rewiring in biological systems is not well understood, but it is believed that the two processes work together but operate on different timescales with weight changes being on a faster timescale than rewiring (Chklovskii et al., 2004). Nothing was added to the rewiring scheme to enforce a

difference in timescale.

Parameter	Value (16x16 Motor System)	Value (116x116 Visual System)
MaxConnE (max connections exc)	30	30
MaxConnI (max connections inh)	120	800
W_{lat}, lateral synaptic weights	Randomly initialised between 0.3 and 0.4 (exc) and -0.3 and -0.4 (inh)	Randomly initialised between 0.3 and 0.4 (exc) and -0.3 and -0.4 (inh)
Exc_p_{conn}, connection probability for lateral excitatory connections	Calculated as *exp(-dist/sigma)* where dist is the Euclidean distance between the neurons and sigma is 3.5	Calculated as *exp(-dist/sigma)* where dist is the Euclidean distance between the neurons and sigma is 3.5
Inh_p_{conn}, connection probability for lateral inhibitory connections	Calculated as *exp(-sigma/dist)* where dist is the Euclidean distance between the neurons and sigma is 8.0	Calculated as *exp(-sigma/dist)* where dist is the Euclidean distance between the neurons and sigma is 8.0
PruneThresh	< 0.3 (exc) , > -0.3 (inh)	< 0.3 (exc) , > -0.3 (inh)

Table 4- Summary of rewiring model parameters

5.4 Conclusions

A full description of the experiments and detailed results are described in Chapter 9 and a results discussion is given in Chapter 11. A summary of the main features of the results is given below.

- The Plasticity Resource works as required for both motor (16x16 cortex and 48x48 cortex) and visual map learning and achieves its purpose of a more autonomous monitoring and control of map development.
- Using online random pattern presentation plus the Plasticity Resource allows monitoring and control of learning for a new dataset where the number and composition of input patterns required is unknown.
- The rewiring scheme is tested with both motor and visual map development and shows that the map results are qualitatively similar to the original systems and

the connectivity in the final networks is sparser than without rewiring.

- Although the overall effect of rewiring is to reduce connectivity, the experimental results showed a high level of both pruning and creation during the course of training, which is in agreement with what has recently been found experimentally.

6 Sensorimotor Integration

6.1 Introduction

The core content of this thesis has been presented in Chapters 3-5: a framework for the development of separate motor and visual cortical maps plus adaptive, autonomous control of the map development process. This chapter takes a slight digression to consider how it might be possible to achieve embodied neural-motor integration in a simplified environment in advance of implementation on a real robot. In particular, asking such questions as what the practical issues are for integration, how is the training process performed and how might at least part of a sensorimotor loop be implemented, i.e. sensory input →neural learning→ motor action. Section 6.2 describes a preliminary modelling study following the Computational Neuroethology ethos: how a real animal sensorimotor behaviour system has been created by combining a neural model with a motor simulation. Section 6.3 builds on this by considering how the motor cortical feature map development system described in Chapter 3 has been integrated with a humanoid robot simulation, both to provide a training environment and as a means to test how the response of the trained motor map can be used to produce a motor movement. Section 6.4 concludes the chapter by summarising the main points and results.

6.2 Prototyping Sensory, Neural and Motor Integration

Rationale

Chapter 2 noted the importance of Computational Neuroethological modelling in that it can provide stepping stones to achieve more complex bio-inspired models on robots ('Neurorobotics'). In the current work it has not been possible to attempt a full implementation on board a robot due to the time needed to develop the theory and computational models. However, it was considered essential to do at least some general

work in the area of sensory-motor integration. To this end a small Computational Neuroethological modelling study was done using the software packages employed in this research to demonstrate a practical integration of spiking neural network modelling and physics simulation of motor action. The main aim was to better understand the issues involved in translating signals generated from a simple spiking neural network into movement of a simulated robotic agent. This work has been published in Adams et al. (2010; 2011) and copies of the papers are included at the end of this thesis.

Modelling prey orientation detection in arachnids

The work of Brownell and collaborators examined the orientation behaviour of the Desert Scorpion, *Paruroctonus Mesaensis* which is nocturnal and able to orient towards prey purely by detection of vibrations carried by the sand substrate (Brownell, 1977; Brownell and Farley, 1979). The vibrations are picked up by detectors called basitarsal compound slit sensilla (BCSS) which are present on the tarsi of the scorpion's eight legs. A mathematical model described in Stürzl et al. (2000) and Brownell and van Hemmen (2001) was based upon the findings of this experimental work and was able to reproduce similar results to those seen in the real animal. The neural model consists of a ring of eight sensory spiking neurons representing the basitarsal compound slit sensilla (BCSS) mechanoreceptors present on each of the arachnid's legs. In the real animal the legs are held in a 'ready' stance at specific orientations relative to the body (+/- 18˚, +/- 54˚, +/- 90˚, +/- 140˚). These sensory neurons are linked with excitatory connections to eight command neurons that represent control structures in the Sub-Oesophageal Ganglion (SOG), a major component of the nervous system in arachnids. Each BCSS / command neuron pair is linked to an inhibitory interneuron. Figure 22 illustrates the arrangement and connectivity of neurons. For clarity, only connections through three legs and one interneuron are shown.

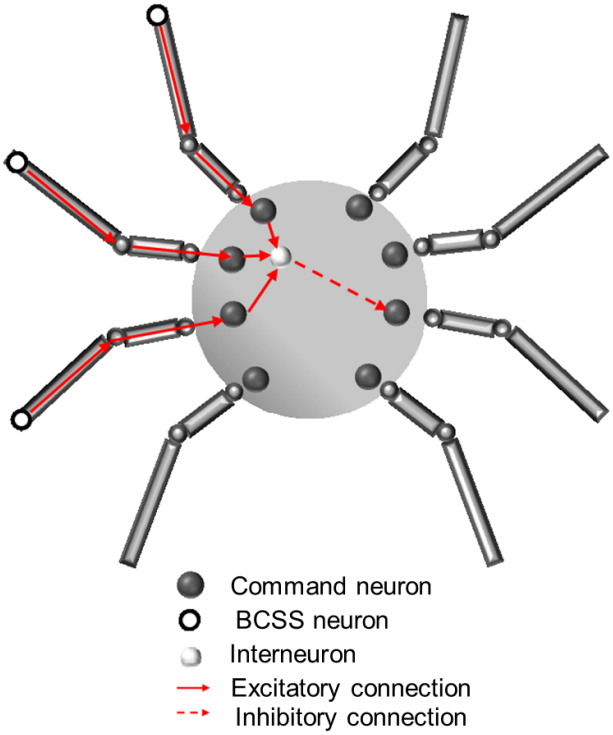


Figure 22-The Arachnid neural Network
(reproduced from Fig 1, Adams et al., 2011)

Command neurons connect in 'triads' to inhibitory interneurons (Figure 22 illustrates one such triad), which are in turn connected to a command neuron on the opposite side of the network. The placement of legs, and thus sensors at intervals around the body determines the information available to the arachnid to enable it to estimate the prey orientation: the crucial information is actually the delay between activation of the sensors of each leg as the wave signal arrives. As shown in Figure 22 each command neuron receives both excitatory and inhibitory signals from BCSS sensory neurons. Excitatory signals come from the BCSS neuron directly linked to a command neuron and inhibitory signals come from the inhibitory triad on the opposite side of the network. The time-window for activation of a command neuron depends upon the delay between activation and inhibition and the number of spikes generated depends upon the length of the time window in which the signal is received. Command neurons at or near the prey orientation will in general receive more spikes as excitatory signals from the command

neurons are not inhibited by the opposing interneuron quickly enough. Similarly, neurons on the opposite side to the prey will be inhibited more quickly by the firing command neurons on the side of the prey and so produce less spikes. According to Brownell (1977) and Brownell and Farley (1979) the BCSS mechanoreceptors in the real animal are specifically activated by Rayleigh (surface) waves travelling through the sand. Using the physical characteristics of Rayleigh waves in sand from Brownell (1977) the neural model represents the wave signal mathematically as a discrete Gaussian distribution of cosine waves. The simulation output is a vector of spike counts from the neurons corresponding to the eight legs and this information can be used to estimate the prey orientation by using a standard population vector decoding technique such as described in Georgopoulos et al. (1986). The orientation neural model of Stürzl et al. (2000) had previously been implemented in the Brian spiking neuron simulator (available for download as a code example) and was used as the basis of the neural model in the current work.

Modelling a Robotic Arachnid

In reality, the arachnid leg is complex and has several segments and joints. For the purpose of the current work this has been simplified and only two segments and two joints per leg were used to control the 'swing' (forward-backward) and 'stance' (up-down) phases of locomotion. The walking gait used by real arachnids is described in Shultz (1987) and Root (1990) and can be approximated by a tetrapod walk where exactly four legs are on the ground and four off at any one time. This is usually an L1, L3, R2, R4 / R1, R3, L2, L4 pattern.

For implementing the arachnid in simulation freely available software has been used which interfaces easily with the Brian Python code. For initial prototyping, PyODE (a

Python interface to the physics simulator Open Dynamics Engine)[4] and VPython, a Python visualisation module[5] were used to create the robotic arachnid simulation. However there were some issues with the performance of the motor parts of this model which were attributed to the physics simulation implementation. Although the PyODE wrapper was easy to use it transpired that ODE required careful parameter tuning to get a stable simulation. Even though some tuning was subsequently done, it was not possible to get completely satisfactory motor behaviour. The results of this initial work are described in Adams et al. (2010). The motor model was subsequently improved by using a different physics simulation based upon the commercial physics engine Nvidia PhysX with the JPhysx Java wrapper[6] (see Adams et al., 2011 for details).

The legs have been modelled in the physics simulation with simple 1 Degree Of-Freedom (DOF) hinge joints. Movement is controlled by sinusoidal generators which calculate the 'set point' or angle for each 1 DOF joint at each time step. These are of a form similar to that used in Crespi et al. (2005) for biologically inspired snake robots. The amplitude of the sinusoid controls the extent of swing of the joint up and down or side to side and is measured in degrees. The frequency sets the number of swing cycles executed per second. These parameters are set at predefined values that give a reasonable height of step and speed of movement. The phase parameter is an offset to control how each joint executes the swing cycle relative to the other joints and is measured in cycles. To implement the tetrapod gait pattern, the phase parameter for swing and stance joints is set so that adjacent legs cycle out of phase. Legs on the left and right sides of the body also operate out of phase. Figure 23 shows a diagram of the simulated arachnid body and leg arrangement. The main body consists of four spheres

[4] http://pyode.sourceforge.net/

[5] http://vpython.org/

[6] http://developer.nvidia.com/page/home.html and http://sourceforge.net/projects/jphysx/

linked by rigid joints with a pair of legs attached to each sphere. The 'head' end of the arachnid is the sphere holding legs 1 and 8.

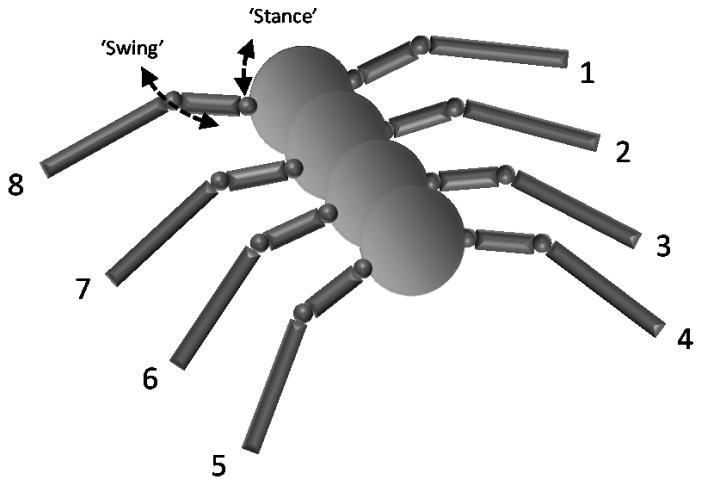


Figure 23 – The Arachnid Model
(reproduced from Fig 2, Adams et al., 2011)

Figure 24 shows a snapshot of the constructed arachnid at its starting point in the simulated world. The arachnid starts the simulation aligned along the positive x axis and with the head placed at the centre of the world (0,0,0). The x axis is designated to be 0˚ with movements clockwise from this point being positive angles and those anticlockwise being negative angles. In the simulation 1 distance unit is taken to be 1 cm. The prey is constructed as a simple white sphere and is positioned at a random angle up to +/- 180 degrees with respect to the initial arachnid position. Please note that here the term 'prey angle' or 'prey orientation' refers to this angle of the prey with respect to the arachnid. The prey is not able to move during the simulation.

Integration of the Neural and Motor Systems

The original orientation neural model successfully generates a prediction of the prey angle with accuracy comparable to the performance of real arachnids (see Stürzl et al., 2000; Adams et al., 2011 for a discussion). However, there is little or no precise information in the literature about how this information might be used in real animals to

cause orientation towards the prey, or how the mechanism might be used to generate an orientation sensing behaviour for a robot. Likewise there has not been any previous work which attempts to model the motor control of arachnid prey localisation.

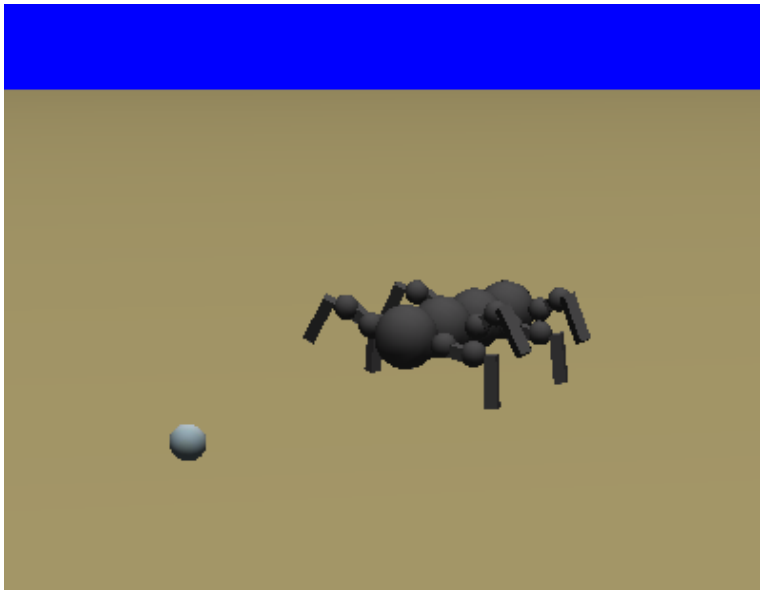


Figure 24 – The Simulated World Showing Arachnid and Prey

Consequently, a main aim of this study was to extend the arachnid prey orientation neural model to include motor behaviour. In particular, to demonstrate how sensory input (vibration waves caused by moving prey) could be linked to motor output (predator orients and moves towards prey) via a biologically realistic spiking neural network.

Although a motor model was not developed in their work, the final discussion section of Brownell and Farley (1979) considers the turning behaviour of spiders and mentions the results of some previous work in Land (1972) which examined walking and turning behaviour in real spiders. This work showed that turning is a very simple modification of the standard walking gait. Legs that move in phase and in the same direction on opposite sides of the body during walking (i.e. the L1, L3, R2, R4 / R1, R3, L2, L4 pattern described in Section 6.2.3) stay in phase but step in *opposite* directions during turning. It also confirmed that stepping frequency and amplitude on both sides remain

the same during turning as for normal walking. Given this situation, it is likely that there is a Central Pattern Generator (CPG) system governing the basic gait, but this can receive an overriding tonic signal that temporarily changes the pattern to produce the turning movement. Such systems are ubiquitous in nature and have been extensively studied in the field of biologically inspired robotics; for an example see Ijspeert et al. (2007). Therefore this approach was used in the current work. The basic tetrapod gait described in Section 6.2.3 is used as the default walking pattern and is modified by a 'tonic signal' to change to turning behaviour. In this work the sign and magnitude of the angle calculated by the neural processing are used as the tonic signal.

The integrated system is setup such that the neural processing is initiated first (from the random generation of a prey angle) and runs on its own processing thread. Concurrently, the motor simulation runs and the arachnid assumes the 'ready' stance. Once the neural processing has generated the result (an angle specified as a value + or – from the 0˚ position) it communicates this to the motor thread. The sign of the angle is used to directly modulate the sign of the signal generated by the sinusoidal controllers and sets the turning direction. The magnitude of the angle is translated into a number of time steps of the physics simulation that should be spent turning. This involves simply multiplying by a factor that was determined by tests with the simulator to determine how many time steps were needed to achieve a turn of a specified angle. It should be noted that once the sensory processing has communicated back its information there is no mechanism for feedback from the motor behaviour to the sensory processing. The simulation ends once the motor behaviour has been completed.

Rationale for an arachnid distance estimation model

According to Brownell and van Hemmen (2001) the desert scorpion has two distinct behaviours, which involve an orientation response. These are the Defensive Orientation Response (DOR), where the animal orients only and the Predator Orientation Response

(POR) that involves orientation and movement towards the prey. The latter behaviour allows real scorpions to accurately catch prey in one movement if it is within 20 cm radius. It is reasonable to assume that distance estimation must be an important element of this behaviour in order to vary the amount of forward movement with prey distance. Although distance sensing is mentioned in some of the orientation sensing works there does not seem to have been any research investigating the neural mechanism in real scorpions or any theoretical or software models of such a process. Therefore, an original contribution of this modelling study was to propose a possible mechanism, involving a simple modification of the existing orientation neural model.

A prey animal moving along on the ground produces both surface transverse travelling waves (Rayleigh waves) and longitudinal travelling waves (P waves). The P waves travel approximately three times faster than the surface waves according to Brownell (1977). In theory then, an arachnid that can detect both of these types of wave could judge the distance of the prey by detecting the difference in arrival time for the two waves (Stürzl et al., 2000; Brownell and Farley, 1979; Kim, 2006). Brownell and Farley also proposed that the varying amplitude of P waves across leg sensors could be used as P waves dissipate more quickly than Rayleigh waves over distance. Results from the experimental work with the Desert Scorpion suggested that hairs present on the bottom of the tarsi which are directly in contact with the sand are the main mechanoreceptors for detecting P waves; the basitarsal compound slit sensilla (BCSS) appear to perform *only* surface (Rayleigh) wave detection. In the orientation model of Stürzl et al. (2000) an approach was used where command neuron activation is determined by the balance between excitation from BCSS sensors in the direction of prey and inhibition from BCSS sensors on the opposite side. A similar principle can be used to estimate the prey distance by using the interaction between P and Rayleigh waves reaching the BCSS and tarsal hair sensors respectively. For instance, as the P

waves are fastest they should arrive first and activate the tarsal hair sensors of all legs. Upon later arrival of Rayleigh waves, activation of the BCSS sensors could be used to inhibit tarsal sensors thus closing the 'window' of activity. The amount of tarsal hair sensor activity during this window would thus encode the distance to prey and govern the strength of the forward response. According to Brownell (1977), the P wave sensor mechanism in scorpions is fairly short range and operates best up to about 10 cm whereas the Rayleigh waves can be detected up to 50 cm. In the current study this distinction is not made, and both waves are assumed be detectable over the same range.

The distance estimation neural model

Following on from the ideas discussed in the previous subsection, the original neural orientation model was extended to include eight tarsal hair sensor (THS) neurons and their corresponding command neurons. These have the same synapse connection structure between them as the BCSS sensory and command neurons shown in Figure 22. In addition the THS and BCSS sensory neurons in each leg are connected by strong inhibitory connections so that as soon as BCSS neurons fire they drastically reduce the activity of the THS neurons. The P wave has been modelled using the same method as the Rayleigh wave (a discrete Gaussian distribution of cosine waves) but instead using information about P waves obtained from Brownell and van Hemmen (2001). As the wave signals over time are generated by a computational model and do not actually travel, the speed difference between P and Rayleigh waves has been simulated by altering the time constant of the model equations for the tarsal hair sensors and their command neurons so the response compared to the BCSS sensors is proportionately faster dependent on the distance between arachnid and prey. It has been estimated elsewhere that the time difference is of the order of 1.3 ms per 10 cm distance (Brownell and Farley, 1979) and so this factor is used in the current work.

The arachnid model already had a basic walking mechanism so it was necessary to

complete the motor model by creating a mechanism to predict the prey distance and convert this to a walking motion. In contrast to the orientation sensing model, which uses a population vector decoding method to estimate the prey direction, the distance neural model uses the total activity generated by the tarsal hair sensors (i.e. the sum of spikes from all legs) over a simulation time of 500 ms. Using a standalone version of the extended neural model, test runs were done to establish the quantitative relationship between the THS activity and distance to prey so that this could be used to predict distance based upon neural activity. The total numbers of THS and BCSS spikes generated during a 500 ms run were collected for a range of distances from 0.1-0.6 metres. Figure 25 shows a plot of activity (total spike count over 500ms) vs. distance for both BCSS and THS sensors.

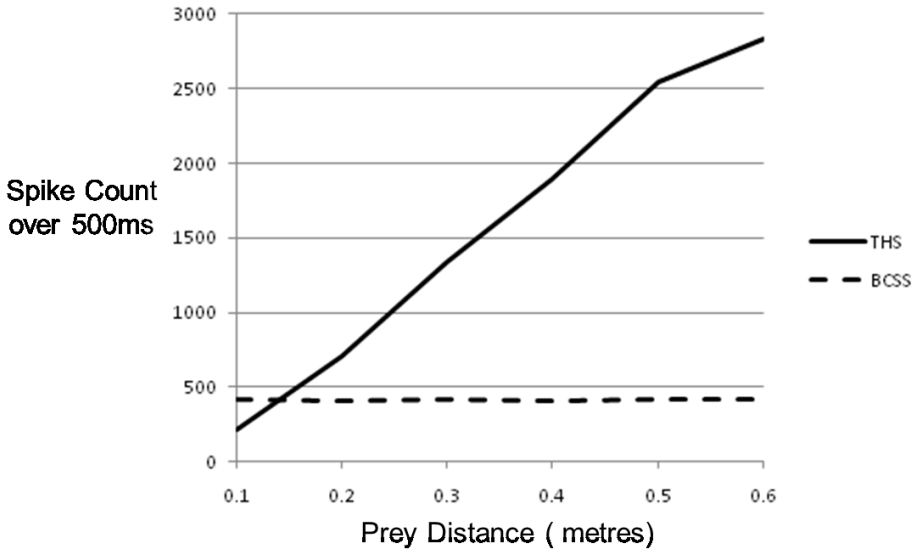


Figure 25 – BCSS and THS sensor activity with prey distance
(reproduced from Fig 4, Adams et al., 2011)

This graph shows that the total activity of BCSS sensors in this model (dashed line) is not distance sensitive (the orientation sensing relies only on timing differences in activity between legs). However, there is a distinct relationship between prey distance and THS activity (solid line). The greater the distance between the arachnid and prey,

the bigger the time window for fast travelling P waves to activate the THS sensors and so more spikes are generated. At close distances, THS activity is almost instantaneously suppressed below the activity of BCSS sensors due to the inhibitory nature of the connections between them.

The relationship shown by the solid line in Figure 25 is used to estimate the prey distance based upon the THS spike count returned from the neural processing. The distance prediction is then translated into a number of physics simulation time steps to be spent walking. This involves simply multiplying by a factor that was determined by tests in the simulation measuring how many time steps were needed to walk a fixed distance. In the extended motor model, the turning and walking behaviours are combined in a simple way. Turning behaviour is executed first to face the direction of prey and then walking behaviour is executed to walk towards it, i.e. turning/walking behaviour is not concurrent. Although this was done for simplicity, it was later realised that according to Brownell and Farley (1979) this kind of separation of the two behaviours has actually been observed in real arachnids!

Validating the Models

Once the orientation neural model had been extended with the distance sensing mechanism and motor behaviours added via the physics simulation, 100 trials of the software were run recording the turning and walking behaviours in response to a prey placed randomly at an angle up to +/- 180 degrees orientation and 8 – 20cm distance with respect to the arachnid starting position. The orientation and distance performance are discussed separately in the following subsections.

Orientation Performance

The actual prey angle, estimated prey angle from the neural model and final angle of the simulated arachnid were collected over 100 trials. Figure 26a shows a comparison of actual prey angle against the prediction of the neural model thus giving an indication of

the performance of the neural model only. Figure 26b shows a comparison of the neural prediction against the final arachnid angle and so indicates the performance of the motor component of the system in executing the required turn. Both graphs show a clear straight-line relationship with data points spread evenly across the linear regression line. In terms of the neural model alone (Fig 26a) the average absolute error (with respect to the actual prey angle) over the 100 trials is 6.52 degrees. In terms of the combined neural and motor model (Fig 26b) the absolute average error (with respect to the actual prey angle) is 15.47 degrees. Therefore, the neural model is capable of a very accurate prediction of prey orientation but some accuracy is lost during actual movement. According to the graph in Fig 3. (a) in Stürzl et al. (2000) the error in the real scorpion is in the region of +/- 12 to 15 degrees. Therefore the integrated system behaviour is close to the performance expected of the real animal.

Distance Sensing Performance

The actual prey distance, estimated prey distance from the neural model and final distance travelled by the simulated arachnid were collected over 100 trials. Figure 27a shows a comparison of actual prey distance against the prediction of the neural model thus indicating the performance of the neural component only. Figure 27b. compares the neural prediction against the final arachnid distance and so indicates the performance of the motor component of the system in executing the walk to reach the prey. Again, both graphs show a clear straight-line relationship with data points spread evenly across the linear regression line. In terms of the neural model alone (Fig 27a) the average absolute error (with respect to the actual prey distance) over the 100 trials is 1.00 cm. In terms of the combined neural and motor model (Fig 27b) the absolute average error (with respect to the actual prey distance) is 1.25 cm. Although there are no results from a real animal to compare to in this case, this seems a reasonable level of performance given the distance over which the prey can be sensed (20 cm) and the size of the scorpion (2.5

cm).

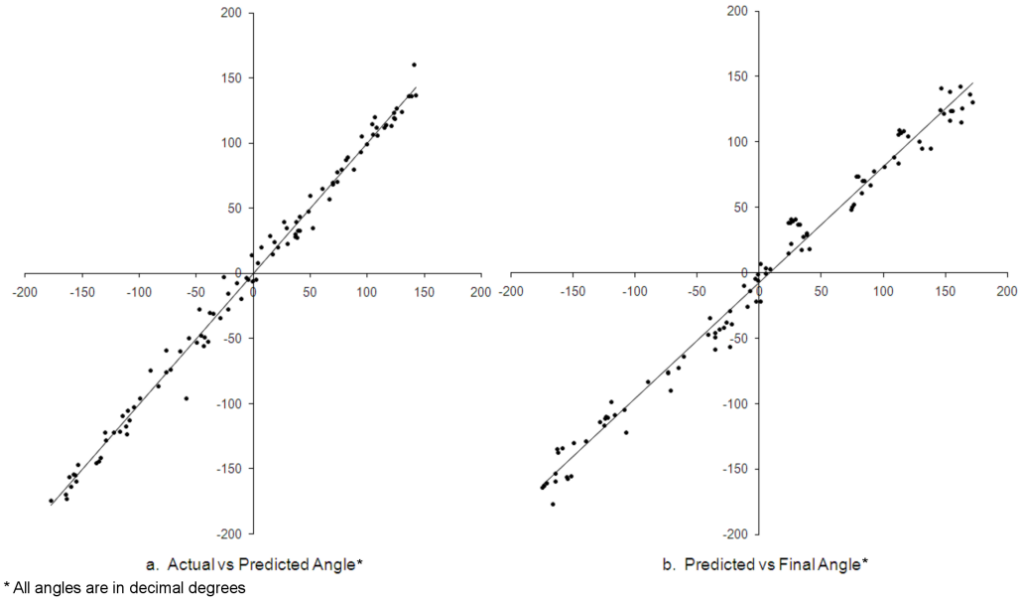


Figure 26 – Performance of the neural and motor orientation models
(reproduced from Fig 5, Adams et al., 2011)

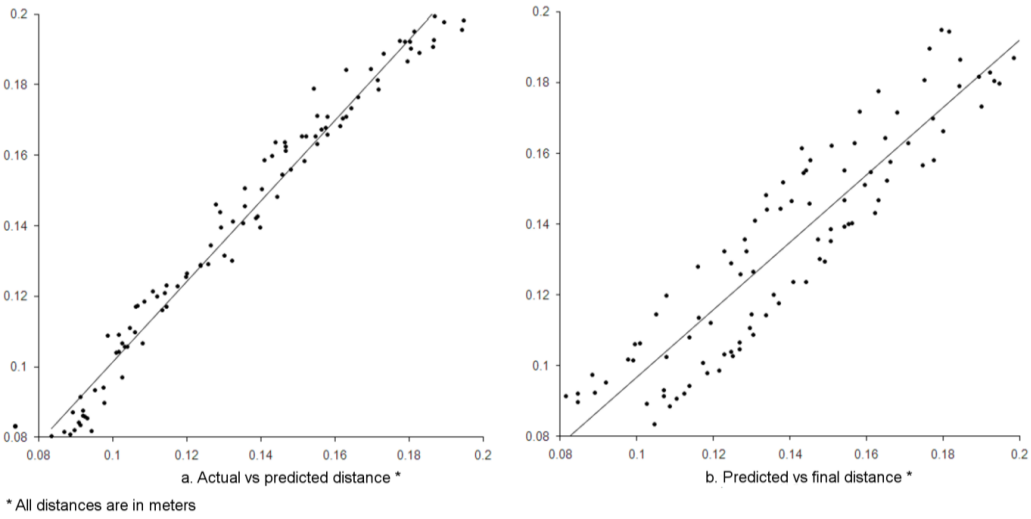


Figure 27 – Performance of the neural and motor distance models
(reproduced from Fig 6, Adams et al., 2011)

Conclusions

As well as being a successful independent study in its own right, the work described in this section prototyped several aspects of neural-motor integration that were fed into the subsequent work integrating motor cortical feature map training with a humanoid

simulation (see Section 6.3). The relevant points are summarised below:

- As envisaged in Chapter 2, it has been demonstrated that it is straightforward to integrate Brian spiking neural network code with other Python packages to make use of spiking neural processing in hybrid applications. In this case VPython/PyODE for physics simulation and visualisation. It was also useful to discover that it is also possible to integrate Python with other languages in a straightforward way: The JPhysX physics simulator used in the improved version of the arachnid simulation is implemented in Java.

- It became clear that physics simulation tuning can be a problem and it should be expected that some effort and time is required to get robust simulations when creating from scratch. In this study the initial implementation significantly affected the accuracy of the motor system despite the fact that the neural predictions of angle and distance were accurate. In this case switching to a commercial simulator did substantially improve performance.

- Using threading to handle the management of Neural and Motor processing was a technique that worked well as it enabled an existing module (the original neural model) to be encapsulated inside its own process and work alongside other new modules (the physics simulation of the arachnid) whilst still exchanging information.

- In terms of handling the transfer of information from the Neural to the Motor domain, two methods were used. Firstly a population decode whereby a consensus decision from a population was used for the orientation prediction as this was a feature of the existing model. Secondly, a spike rate generated by an entire population over a fixed time was used for the distance prediction. In both cases the prediction was converted to a number of time steps to be spent either

turning or walking for execution directly by the motor simulation.

6.3 Motor Map Learning in a Humanoid Robotic Simulation

Rationale

The study described in Section 6.2 involved a simple hardwired 'brain' with no learning so the next step was to apply some of the knowledge gained from the exercise to integrate the motor cortical feature map development system that forms part of the core of this thesis with a humanoid robot simulation. In particular to explore how to use the output from a trained map to produce a motor movement in response to an input signal based upon the learned repertoire of directions.

CLrobotsim: a Python robot simulator

Fortunately, it was not necessary to create a humanoid robot simulation from scratch. ClrobotSim (Cheung, 2008) is a freely available Python based robotic simulator which comes with a few model examples. One of these implements a model of just the legs of a humanoid robot, similar to the actual servomotor arrangement of a Robotis Bioloid (Robotis website, undated). See Figure 28.

This software has been created using PyODE and VPython and although some issues were found with PyODE in the work described in Section 6.2, in this case it appears that a lot of effort has gone into making this simulator robust with all the simulation parameters being tuned appropriately. The core of the software is the physics simulation loop which contains calls to continuously update objects in the environment. The method of modelling the robot legs has been done in a detailed and modular way. The legs are constructed from simulated servomotors and various types of joints.

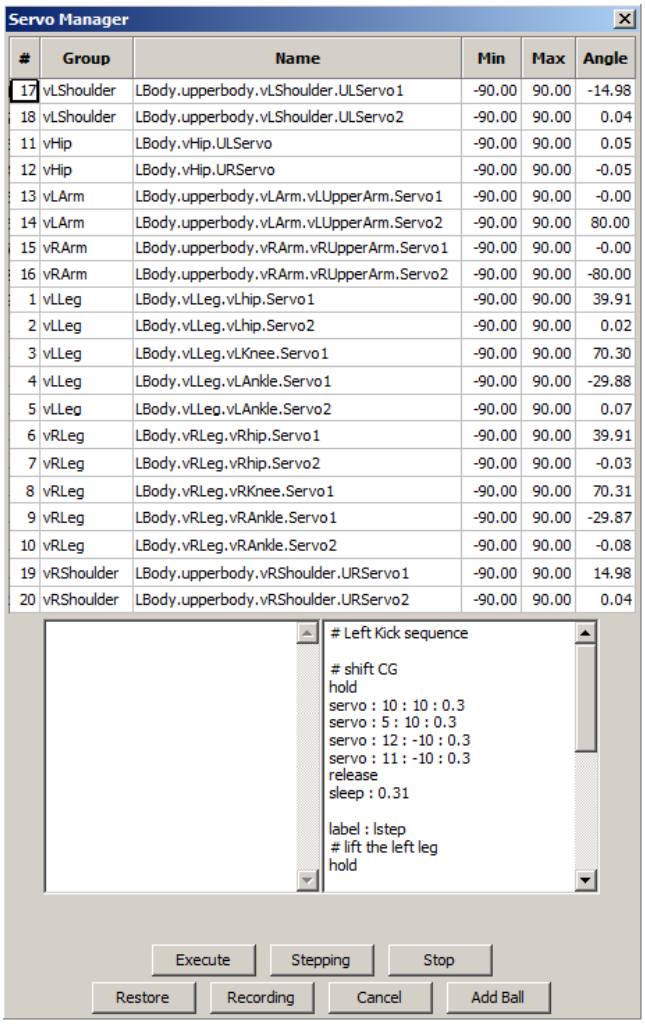
Figure 28 has been removed due to Copyright Restrictions

Figure 28 – The Robotis Bioloid Humanoid Robot

(picture taken from the Robotis website)

Other functionality includes a Servo Manager which gives a continually updated display of the readout of all the servomotors in the simulation. Robot movement sequences can be effected by inputting commands in a simple scripting language. When a movement sequence is executed it runs on a separate thread to the main simulation loop. See Figure 29 for a screenshot of the servo manager and movement sequence input panel.

Due to the modular and Object Oriented design of the core software it was straightforward to create a new simulation model of a complete humanoid robot by adding extra servomotors to make arms, a torso and a head as well as other environment objects for it to interact with. See Figure 30 for a screenshot of the extended robot simulation. Other amendments were made to the software to develop an orientation behaviour for the robot. Basic left and right turn sequences were created whereby one turn moves the robot one compass direction either left or right of its current orientation.



#	Group	Name	Min	Max	Angle
17	vLShoulder	LBody.upperbody.vLShoulder.ULServo1	-90.00	90.00	-14.98
18	vLShoulder	LBody.upperbody.vLShoulder.ULServo2	-90.00	90.00	0.04
11	vHip	LBody.vHip.ULServo	-90.00	90.00	0.05
12	vHip	LBody.vHip.URServo	-90.00	90.00	-0.05
13	vLArm	LBody.upperbody.vLArm.vLUpperArm.Servo1	-90.00	90.00	-0.00
14	vLArm	LBody.upperbody.vLArm.vLUpperArm.Servo2	-90.00	90.00	80.00
15	vRArm	LBody.upperbody.vRArm.vRUpperArm.Servo1	-90.00	90.00	-0.00
16	vRArm	LBody.upperbody.vRArm.vRUpperArm.Servo2	-90.00	90.00	-80.00
1	vLLeg	LBody.vLLeg.vLhip.Servo1	-90.00	90.00	39.91
2	vLLeg	LBody.vLLeg.vLhip.Servo2	-90.00	90.00	0.02
3	vLLeg	LBody.vLLeg.vLKnee.Servo1	-90.00	90.00	70.30
4	vLLeg	LBody.vLLeg.vLAnkle.Servo1	-90.00	90.00	-29.88
5	vLLeg	LBody.vLLeg.vLAnkle.Servo2	-90.00	90.00	0.07
6	vRLeg	LBody.vRLeg.vRhip.Servo1	-90.00	90.00	39.91
7	vRLeg	LBody.vRLeg.vRhip.Servo2	-90.00	90.00	-0.03
8	vRLeg	LBody.vRLeg.vRKnee.Servo1	-90.00	90.00	70.31
9	vRLeg	LBody.vRLeg.vRAnkle.Servo1	-90.00	90.00	-29.87
10	vRLeg	LBody.vRLeg.vRAnkle.Servo2	-90.00	90.00	-0.08
19	vRShoulder	LBody.upperbody.vRShoulder.URServo1	-90.00	90.00	14.98
20	vRShoulder	LBody.upperbody.vRShoulder.URServo2	-90.00	90.00	0.04

Figure 29- The CLrobotsim Servo Manager Panel

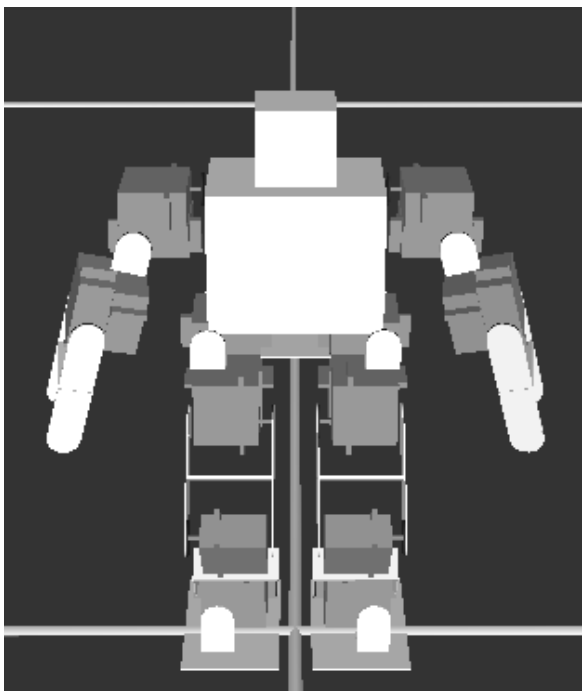


Figure 30 – Simulation of the Full Humanoid Robot

The existing movement generator was amended to be able to make an orientation by passing the left / right sequence plus a number of turns to make. This process is used during testing of the motor map in simulation – see Chapter 8, section 8.4.

Integration of the Neural and Motor Systems

Guided by the information gained from the prototype study of Section 6.2, the neural code developed for the motor map development (see Chapter 3) was treated as an independent module and encapsulated in a separate class in keeping with the Object Oriented implementation of CLrobotsim. The neural object is designed to run as a separate thread from the existing ones in CLrobotsim so that neural processing can happen alongside the normal physics simulation and movement. The humanoid robot model is also a class in its own right and when it is instantiated within the simulation, an instance of the neural class is also created and associated with it.

The first requirement was to extend the simulation software to make a training environment for the development of a motor map. A ‘motor babbling’ process was implemented whereby the robot is made to orient randomly to one of 8 possible directions (N, NE, E, SE, S, SW, W, and NW as for the original motor training). A direction is selected randomly and translated into a sequence of left or right turns needed to orient to that direction which are then executed by the thread responsible for robot movement. When a movement is initiated the neural processing thread is started which results in learning of the corresponding direction. This happens using exactly the same process as described in Chapter 3, Section 3.6. Once the neural processing and movement sequence have been completed the system is allowed to generate another random movement. Once the required number of patterns have been presented the simulation terminates. Various experiments were done with this setup to replicate the basic motor map learning and also with the Adaptive Plasticity methods of Chapter 5. A summary of the highlight results is given at the end of this chapter in Section 6.4 and

the full results are presented in Chapter 8, Section 8.3.

A second requirement for the simulator was to use it as the basis for a testing system whereby the humanoid simulation was instantiated with a trained motor map and is presented with random patterns produced by perturbing exemplars which have not been used for training. The response of the motor map is decoded and the corresponding movement sequence initiated. For example, if the motor map response for pattern 'North' is generated then the robot will turn to face North. Data can then be collected on the actual pattern presented, what was decoded from the motor map response and what movement sequence was executed.

Decoding the Motor Response

So far the issue of how to make use of a trained cortical feature map response has not been discussed. For the motor map, it has been established from plots of the entire maps and of individual responses that there is map organisation and a distinct spatiotemporal response to each pattern after training (see Chapter 8). However, similarly to the Arachnid prototype of Section 6.2, to make use of cortical map responses to produce a specific motor response on presentation of a stimulus pattern requires a decoding of the neural response and translation to a motor behaviour. Therefore a quantitative method of identifying which pattern the cortical response was generated by is required. It is also worth noting that in the current work, the motor input patterns were specially created to have distinctly similar and dissimilar patterns and so it is fairly easy to visually see if the responses are distinct by examining plots of the response. It is unlikely that for 'real world' data there would be such a distinct response. The method used in the current work is based upon the van Rossum metric. Usually this is used in in-vivo work to compare two different spike trains from the same neuron, from different experimental runs and determine if it is in fact the same response within a certain tolerance (van Rossum, 2001). To calculate the metric, firstly the delta function associated with each

spike is replaced with an exponential function as shown in equation (31).

$$f(t) = \sum_{i=1}^{M} H(t - t_i) e^{\frac{-(t-t_i)}{t_c}} \quad (31)$$

Where:

f(t) is a spike train

M is the number of neurons in the spike train

t_i is the spike time for neuron *i*

t_c is a time constant

H is the Heaviside step function (*H(x)* = 0 if *x* < 0 and *H(x)* = 1 if *x* >= 0).

The integral of the difference squared between two spike trains gives a distance measure, as shown in equation (32).

$$D^2(f,g)_{t_c} = \frac{1}{t_c} \int_0^{\infty} [f(t) - g(t)]^2 dt \quad (32)$$

Where:

D^2 is the van Rossum distance metric

f(t) and *g(t)* are two spike trains

t_c is a time constant which defines the tolerance

Figure 31 graphically illustrates the method. In the top half of the figure, two spike trains f and g are convolved with the exponential function and aligned. The t_c parameter controls how much overlap is generated between individual pairs of spikes. In the bottom half of the figure the graph produced by taking the difference squared of f and g is shown and the area under this curve gives the van Rossum metric. Low scores indicate that the distance between spike trains is short and there is a close match while high scores indicate a bigger difference.

Figure 31 has been removed due to Copyright Restrictions

Figure 31 – A Graphical Representation of the van Rossum Metric

(taken from Fig 1. Van Rossum, 2001)

In the current work a modified version of the van Rossum metric is used to compare sets of spikes generated by the cortical map in response to different patterns, and in particular to take into account both temporal and spatial aspects. The procedure is summarised as follows:

1. An input pattern is presented to the trained motor map to generate the response (a set of neurons firing at particular times)
2. The response is compared to a typical response from each of the 8 exemplar patterns
3. For every cortical neuron check whether it has fired in either or both responses:
    a. If it has fired in either or both of the responses then the regular van Rossum metric is calculated using M=1.
    b. If it has not fired in either response it is ignored

4. The scores from step 3a are totalled and averaged
5. The lowest score produced over all the exemplar patterns is determined and this indicates which pattern caused the response

This method implicitly takes into account spatial and well as temporal aspects of the responses as in step 3a the smallest score contributions will come from matching neurons (i.e. the same spatial location and close firing times). In contrast if a neuron fires in only one response the score generated will be high.

6.4 Conclusions

A full description of the experiments and detailed results are described in Chapter 8, Section 8.3 and a results discussion is given in Chapter 11. A summary of the main features of the results is given below.

- Motor map training using the humanoid simulation environment gives qualitatively similar results to the original experiments, although run times are longer due to the simulation overhead.
- Likewise, a version with the Adaptive Plasticity method to control learning gives similar results to the original experiments for the motor map.
- The amended version of the van Rossum metric successfully distinguishes between motor map responses and can be used to make a sensible decision when a pattern is presented that is between two exemplar patterns.
- A training version of the simulation with the van Rossum analysis included demonstrates how accurately the system predicts what the sensory input was and that it executes the correct motor movements. Specific experiments using input that is noisy (i.e. a pattern that is between two exemplars) show that the system makes a reasonable decision and selects a movement that corresponds to the closest exemplar.

7 Coupled Maps

7.1 Introduction

Chapters 3-5 have described the basic SOFM methodology for the creation of separate motor and visual directionally selective maps and also a method of autonomous control of map training. This chapter describes a methodology for coupling these maps together to achieve a simple form of visuomotor coordination which is based upon the methods described in Marian (2002) but with larger maps and visual input from the DVS 128 silicon retina. Section 7.2 describes how the coupling between the two maps is set up and 7.3 the learning regime which includes rewiring plasticity as described in Chapter 5, Section 5.3. Section 7.4 describes the training sequence for coupling and section 7.5 concludes with a summary.

7.2 Coupling Setup

In Marian (2002) a 18x18 visual cortical layer was coupled to a 16x16 motor layer with full connectivity so that the visual neurons had access to the representation of all the directions encoded in the motor map. In the current work a 116x116 visual layer and a 48x48 motor layer are used. See Figure 32 for an overview of the architecture of the coupled system. As the maps here are considerably scaled up from Marian's work initial full connectivity between the maps is not computationally efficient, and actually not necessary as a learning method employing rewiring is used which ensures that new connections can be made as required, dependent upon the correlation of activity in the two maps (see Section 7.3).

For coupling, visual and motor network setup is done exactly as described in Chapters 3 and 4, except that previously saved weight matrices from individual map training experiments are loaded in to set the connection weights and delays. Connectivity between the two cortical layers is assigned randomly with a probability of 0.4, and connection weights are set at random values between 0.4 and 0.5. A summary of the

network and learning parameters are given in Table 5.

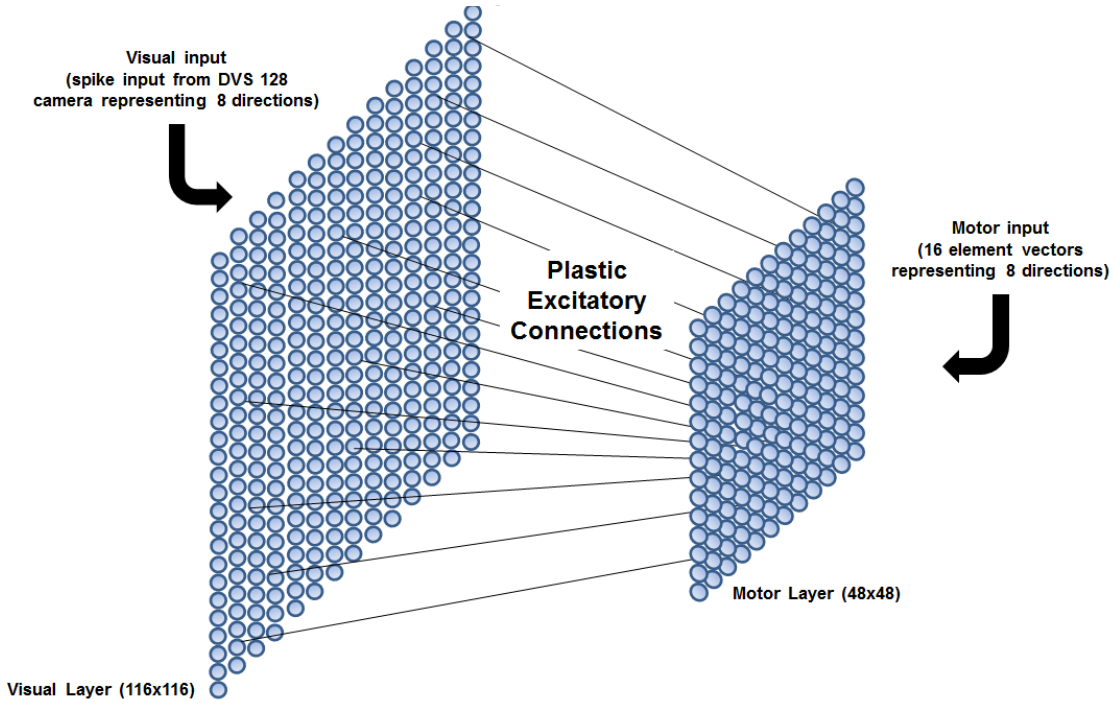


Figure 32- Coupling Architecture

7.3 Learning

Functional Plasticity

Weight changes are controlled by the standard exponential STDP following Song et al. (2000) and the weight dependent methods of van Rossum et al. (2000).

The update rules for Long Term Potentiation (LTP) and Long Term Depression (LTD) are given as equations (33) and (34).

$$\Delta w_{ij} = (w_{ij} \cdot delta_{STDP}),\ w_{t+1} = w_t + \Delta w \quad (33)$$

$$\Delta w_{ij} = 1 + delta_{STDP},\ w_{t+1} = w_t * \Delta w \quad (34)$$

Where:

w_{ij} is the weight on the connection between presynaptic neuron i and postsynaptic neuron j

$delta_{STDP}$ is the standard exponential STDP function:

$$A_p \cdot \exp\left(-\frac{t_j - t_i}{\tau_{ltp}}\right) \qquad (t_j - t_i) > 0$$

$$A_m \cdot \exp\left(\frac{t_j - t_i}{\tau_{ltd}}\right) \qquad (t_j - t_i) < 0$$

t_i, t_j are the firing times of presynaptic neuron i and postsynaptic neuron j respectively.

A_p, A_m are the STDP potentiation and depression rates respectively.

τ_{ltp}, τ_{ltd} are the time constants for potentiation and depression.

See Table 5 for details of the parameter values.

Parameter	**Value**
Network Architecture	
N_v, number of neurons in visual layer	13456 (116x16)
N_m, number of neurons in motor layer	2304 (48x48)
W_{cpl}, synaptic weights	Randomly initialised between 0.4 and 0.5
p_{conn}, connection probability	0.4
STDP Learning	
A_p, LTP rate	0.1
A_m, LTD rate	-1.05 * A_p
τ_{ltp}, LTP time constant	10 ms
τ_{ltd}, LTD time constant	10 ms
Rewiring	
MaxConn (max connections)	850
PruneThresh	< 0.4

Table 5- Summary of coupling network and learning parameters

Structural Plasticity

Rewiring forms an important part of map coupling as it provides an adaptive system that can produce connections where needed and remove unused ones and thus avoids the need for full connectivity between the cortical maps. The rewiring scheme is exactly as described in Chapter 5, Section 5.3 with slight differences in the values for maximum connectivity and pruning threshold as shown in Table 5.

7.4 Putting it All Together

In Marian (2002), the process of visuomotor association was inspired by the 'motor babbling' approach of Kuperstein (1998) which developed a method of achieving visuomotor arm coordination using unsupervised learning. There are two important concepts, firstly:

'*representations of postures emerge out of the correlation between posture sensation and target sensation' (Kuperstein, 1998)*

And secondly:

'motor signals are first generated to explore a large range of arm postures. During each posture, with object in hand, topographic sensory information about the object, projects to a target map through modifiable gating factors, called weights, producing computed motor signals. Differences between the actual motor signals generated for each posture and the computed motor signals are used to change the weights so that these differences are minimized. These weight changes, for all possible motor postures, constitute the sensory-motor correlation and allow the system to become self-consistent.' (Kuperstein,1998)

Essentially what this means is that the random presentation of a 'posture' (motor input) is concurrently presented with the appropriate sensory information (visual input) so that associations between the two are made. In the current work a motor babbling learning cycle similar to that described in Marian (2002) is used, but with amendments to incorporate the spiking visual input from the DVS 128 camera. The sequence is initiated by a random motor movement which activates a population of neurons in the motor cortex. After a short delay the corresponding retinal input commences and the motor cortical response is maintained concurrently at a specific rate. During this period spike-timing dependent learning is applied to the plastic connections between the visual and motor maps. The coupling learning cycle involves the following steps:

1. A direction is randomly selected
2. The appropriate motor input is applied and the motor cortical response is generated
3. The corresponding visual input sequence is applied

4. Motor cortical response is maintained at a rate of 30Hz during step 3.

5. Learning is applied continuously during step 3.

6. On completion of the visual input sequence, the network is reset for the next direction.

7.5 Conclusions

A full description of the experiments and detailed results for coupling are described in Chapter 10 and a results discussion is given in Chapter 11. A summary of the main features of the results is given below.

- Coupling of visual and motor maps results in the visual input modulating the motor response
- Before coupling training the motor output (when controlled solely by visual input) is not particularly directionally selective – there is a lot of overlap in the response to different input patterns. After training a new motor topographic map is established with a distinct spatiotemporal response. In particular the temporal component reflects the temporal characteristics of the visual input that produced it.
- Allowing rewiring results in sparser connectivity than the original setup.

8 Cortical Feature Map Training

8.1 Overview

This chapter presents the results of the motor and visual map training. Section 8.2 describes the validation of what is termed the motor map 'benchmark'. This is the first version of the map training methodology (16x16 cortical size) described in Chapter 3, i.e. the process uses a conventional learning rate parameter and the inputs are predefined training files. The purpose of this experiment was to confirm that a topological map representing the 8 motor directions would be formed and furthermore that the response of the map to different input patterns after training was distinct. This experiment also serves as a control/comparison for later versions of the map training, for instance with the Adaptive Plasticity methods of Chapter 5. Section 8.3 describes a similar validation done for the scaled up 48x48 motor map. Section 8.4 again follows a similar process but this time for validating the training of the 16x16 motor map using the humanoid simulation described in Chapter 6. These latter two experiments are to confirm that changing the scale of the map and the training environment does not affect the results. Section 8.5 validates the development of basic directional selectivity in the Visual map, in particular aiming to reproduce some of the results of Wenisch et al. (2005) and confirm the use of their asymmetric STDP rule but using input from the DVS 128 silicon retina camera. Section 8.6 describes the validation of what is termed the visual map 'benchmark'. This is the first version of the map training methodology described in Chapter 4, i.e. the process uses a conventional learning rate parameter. The purpose of this experiment was to confirm that a topological map representing the 8 visual directions would be formed and furthermore that the response of the map to different input patterns after training was distinct. This experiment also serves as a control/comparison for later versions of the visual map training.

Full discussion of the results is left until Chapter 11.

8.2 Experiment 1 - The Motor Map Benchmark

Experimental setup

The setup as described in Chapter 3 was used. Therefore the learning rules incorporated a traditional learning rate parameter reduction and used predefined input datasets. Table 6 gives a summary of the relevant experimental parameters. A training cycle consisted of the presentation of 1 training dataset. A training dataset consisted of a file containing 20 examples of each of the 8 directional patterns (160 patterns per file) as described in Chapter 3, Section 3.4 and Appendix A. These were generated by randomly perturbing the exemplar patterns. The order of the patterns within the file was also randomised. A different, independently created training dataset was used for each training cycle.

Parameter	Description	Value
Ap	LTP rate	0.1
Am	LTD rate	-0.105*Ap
N	Number of training cycles	10
P	Total number of patterns	1600
η	Initial learning rate	0.5
η_r	Learning rate reduction	0.949

Table 6 - Parameters for Experiment 1

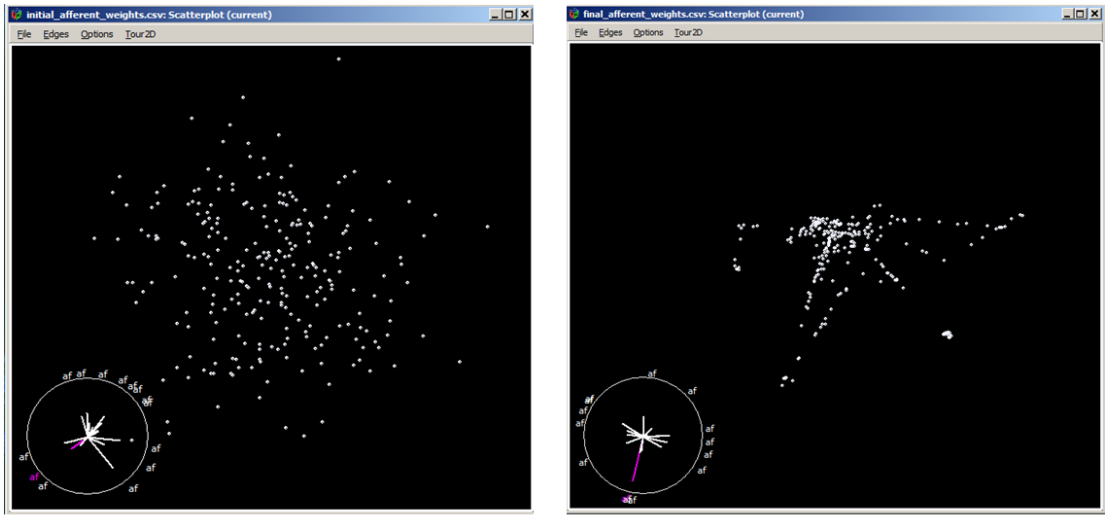
Representation of input vectors by afferent weights

Following the training, two aspects of the afferent learning were investigated. Firstly, whether there had been ordered change in the weights compared to the initial random state. Secondly, to confirm that the afferent weights have learned to represent the input patterns. To get a broad overview of the weight changes a clustering analysis on the afferent weight vectors was performed using the visual tool ggobi[7] which is designed for easy visualisation of high-dimensional data.

Figure 33a shows the results of clustering the 16-dimensional afferent weight vectors

[7] http://www.ggobi.org/

before training and Figure 33b shows the situation after training. Note that these screenshots were taken from a 3D animation rotating through many different views of the data so are only a 2D representation of one view of the structure.



a. Before training b. After training

Figure 33 – Expt 1: Clustering Analysis on the Afferent Weights

It is clear from these pictures that the initial random weights have developed some sort of structure after training and moreover that the structure consists of about 7 or 8 related groups. However it is not clear yet that the structure represents anything meaningful with respect to the input patterns. In order to verify this aspect directly portions of the afferent weight matrices were plotted and compared to the input patterns. Figure 34 shows patches from two completely different parts of the afferent weight matrix before and after training. The box on the right of the figure gives an extract of only the salient portions of some of the input patterns (the parts that make the highest contribution to the PSP – see Chapter 3, Section 3.4). On the left of this figure the weight matrices show no discernible pattern, with all weights falling between their initial values of 0.4 and 0.5. In contrast the weight matrices after training (middle of the figure) show how selected connections have been strengthened and weakened in order to represent the input patterns. The middle plots have been annotated to show where the afferent weights in

these patches have learned to represent specific input patterns.

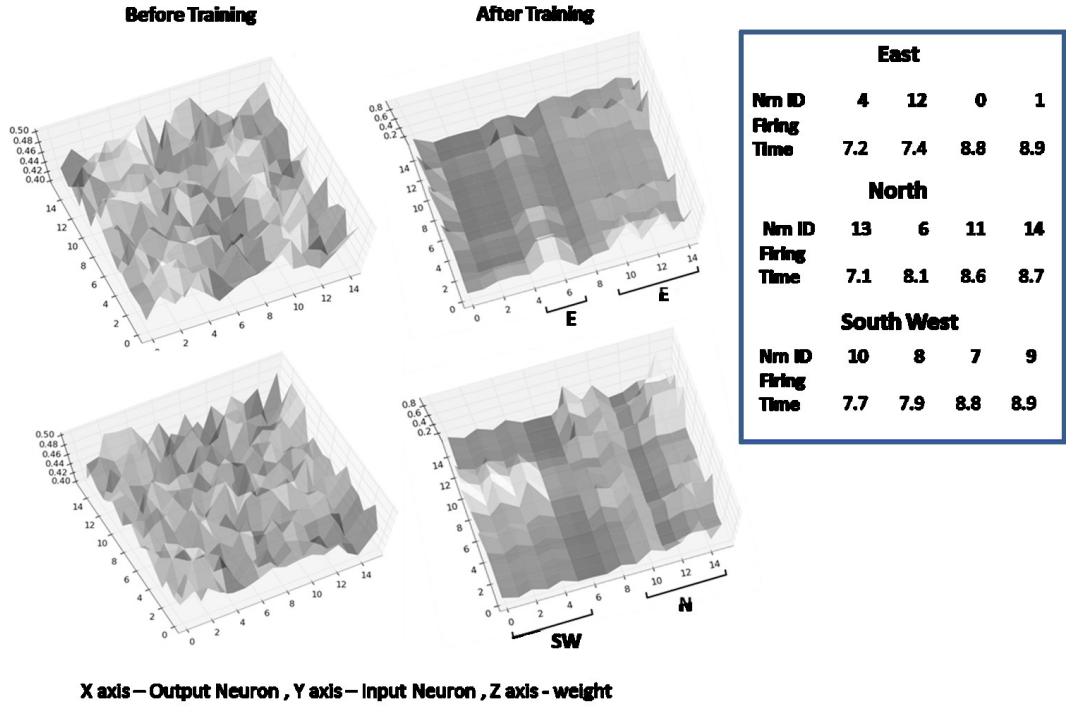


Figure 34 – Expt 1: Representation of the Input Patterns by Afferent Weights

Characteristics of lateral weight training

During training, updates to the lateral weights also occur in response to the spreading of activation amongst neurons in the output layer. The lateral connections are recurrent connections between neurons in the cortical layer and consist of a mixture of short-range excitatory and long-range inhibitory synapses with delays set according to the distance between neurons. Strengthening of excitatory lateral connections results in clusters of neurons located together firing for the same or similar input. Strengthening of inhibitory lateral connections results in clusters of neurons with similar preference inhibiting the action of clusters with a different preference. Figure 35 shows extracts from the excitatory and inhibitory lateral weight matrices before and after training.

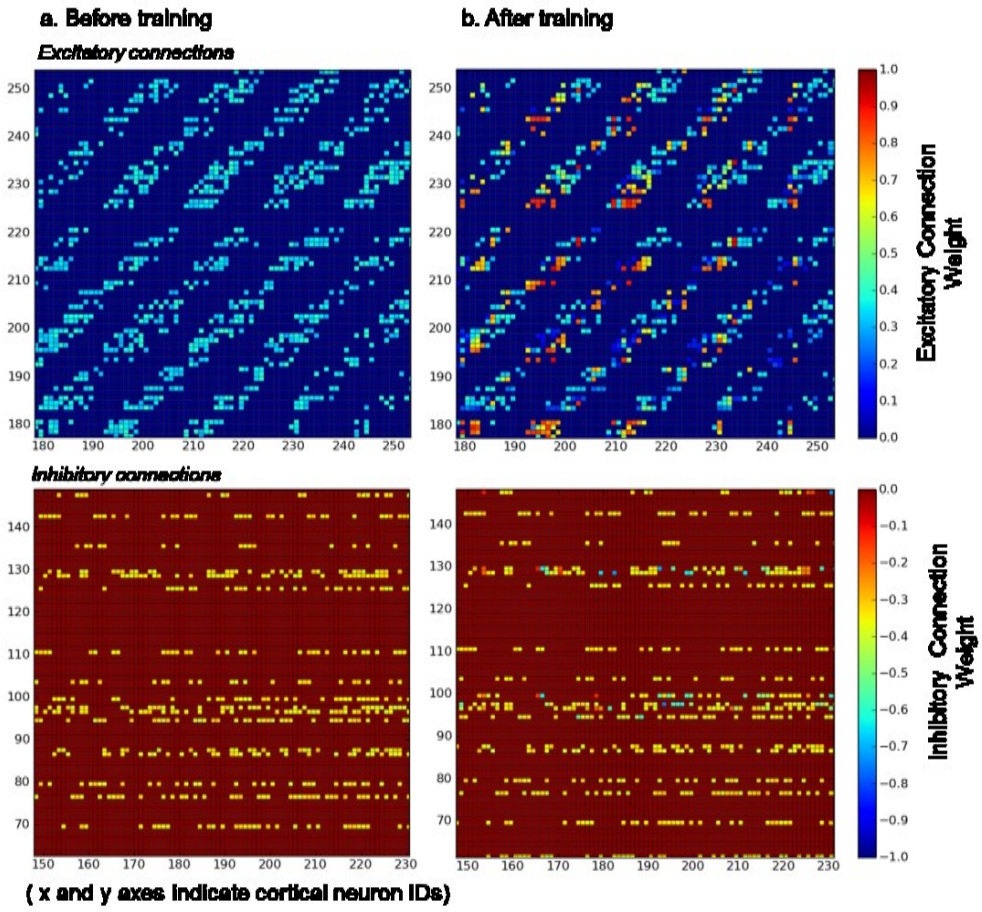


Figure 35 – Expt 1: Lateral Weights Before and After Training

The left hand column (Fig. 35a) shows the case before training and here all weights are at their initial values of between 0.3 and 0.4 (excitatory) or -0.3 and -0.4 (inhibitory). The right hand column (Fig. 35b) shows the changes that have occurred during training. The plot at top right indicates strengthening (orange-red) and weakening (dark blue) for the excitatory connections and there is some evidence of a distance dependent pattern. Neurons in clusters have increased their connection strengths between one another, but moving further out, connections have weakened. The plot at bottom right shows some strengthening (dark blue) and weakening (orange-red) of inhibitory connections.

Spatiotemporal response

In the analysis of regular SOM maps, usually it is the afferent weights that hold all of the information about the learned patterns and moreover the response to a particular

pattern is the activation of single node in the output layer. In the case of real cortical maps, the response to a stimulus is more complicated. They exhibit a distributed response of a population of neurons and also individual neurons may have a preference that is finely or quite broadly tuned. In the case of the current work the cortical map layer has a more complicated response than a traditional SOM due to the fact that there is lateral connectivity, the neurons are spiking neurons and thus convey information by their time of spiking: the learning rules described in Chapter 3, Section 3.5 include a temporal neighbourhood which rewards neurons that fire close to the time of the winner. Therefore it was expected that after training the map layer should have organised its responses topologically but also temporally. Figure 36 shows a plot of the spatiotemporal response of the output layer following the presentation of two patterns expected to be dissimilar (North and South) before and after training.

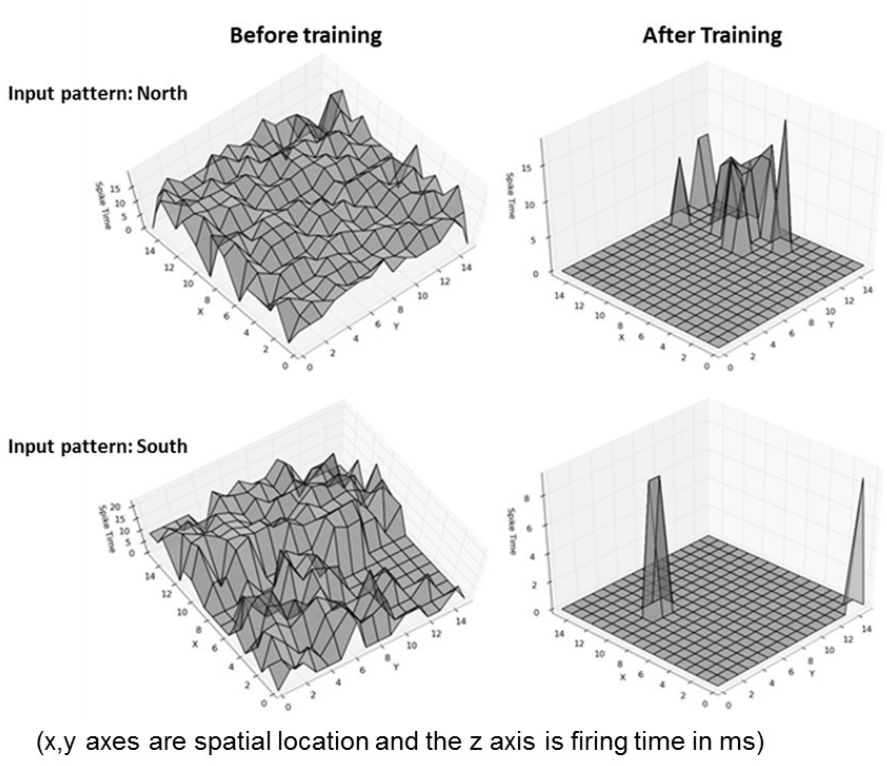


Figure 36 – Expt 1: Cortical layer response to two different patterns before and after training

The untrained network (left hand column) is fairly equally responsive to both patterns

with a large proportion of the neurons firing at various times over almost the whole output period (which is up to 30ms). After training (right hand column) the majority of neurons are silent (due to lateral inhibition from the responding neurons) and a spatially distinct patch of neurons respond to each of the patterns. The ranges of firing times of the responding populations are also different to the initial state. For example, in the response to pattern North the cortical neurons fire between 20 and 25ms whilst for pattern South they fire around 8ms.

The Final Map Topography

Finally, it remains to confirm whether there has been a topological ordering during training, i.e. if a cortical 'map' has formed which represents the 8 directions in the input patterns. Figure 37 shows the composite response of the motor map before and after training in spatial terms only for clarity.

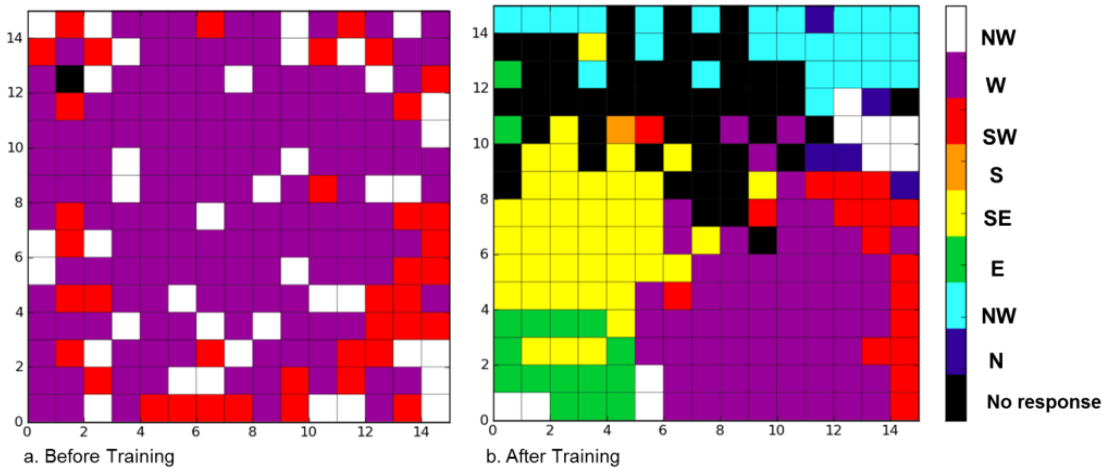


Figure 37 – Expt 1: 16x16 Motor Map Topography

In Figure 37a (map before training) the responses to all the patterns overlap considerably showing no particular spatial ordering. However, in Figure 37b (map after training) over 90% of the cortical neurons have developed a preference for at least one of the 8 directions (there is still some overlap in the response) and neurons with the same preference are mainly grouped in the same spatial location. In the main, neighbouring patterns are also located near to each other, for instance patterns W

(purple) and SW (red) are next to each other as are patterns E (green) and SE (yellow).

8.3 Experiment 2 - The Scaled Up Motor Map

Experimental setup

Table 7 gives a summary of the relevant experimental parameters. Essentially these were the same as for Experiment 1, but less training cycles were used due to the fact that training time was longer with a larger map.

Parameter	Description	Value
Ap	LTP rate	0.1
Am	LTD rate	-0.105*Ap
N	Number of training cycles	5
P	Total number of patterns	800
η	Initial learning rate	0.5
η_r	Learning rate reduction	0.949

Table 7 - Parameters for Experiment 2

Representation of input vectors by afferent weights

As before a clustering analysis on the afferent weight vectors was performed using ggobi. Figure 38a shows the results of clustering the 16-dimensional afferent weight vectors before training and Figure 38b shows the situation after training. The results are similar to those for the 16x16 motor map shown in Figure 33 (except there are more data points) and again show that some sort of non-random organisation of the weights has occurred during training.

Characteristics of lateral weight training

Figure 39 shows extracts from the lateral weight matrix for the excitatory and inhibitory connections for the scaled up motor map before and after training. Qualitatively, the results are the same as for the 16x16 version. Figure 39a shows the case before training and here all weights are at their initial values of between 0.3 and 0.4 (exc) or -0.3 and -0.4 (inh). After training, Figure 39b shows differential strengthening (orange-red for

excitatory connections cyan-blue for inhibitory) and weakening (dark blue for excitatory and orange-red for inhibitory).

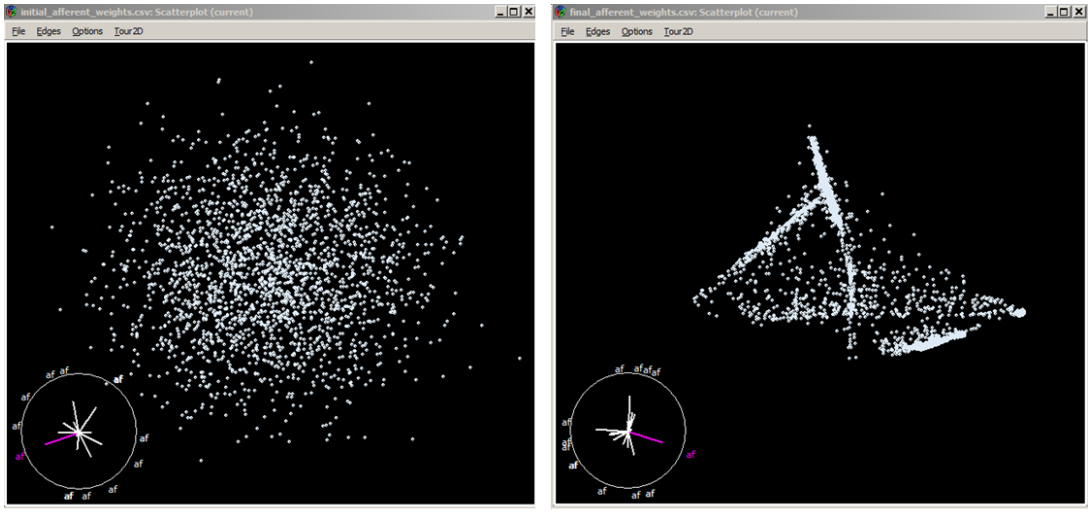


Figure 38 – Expt 2: Clustering Analysis on the Afferent Weights

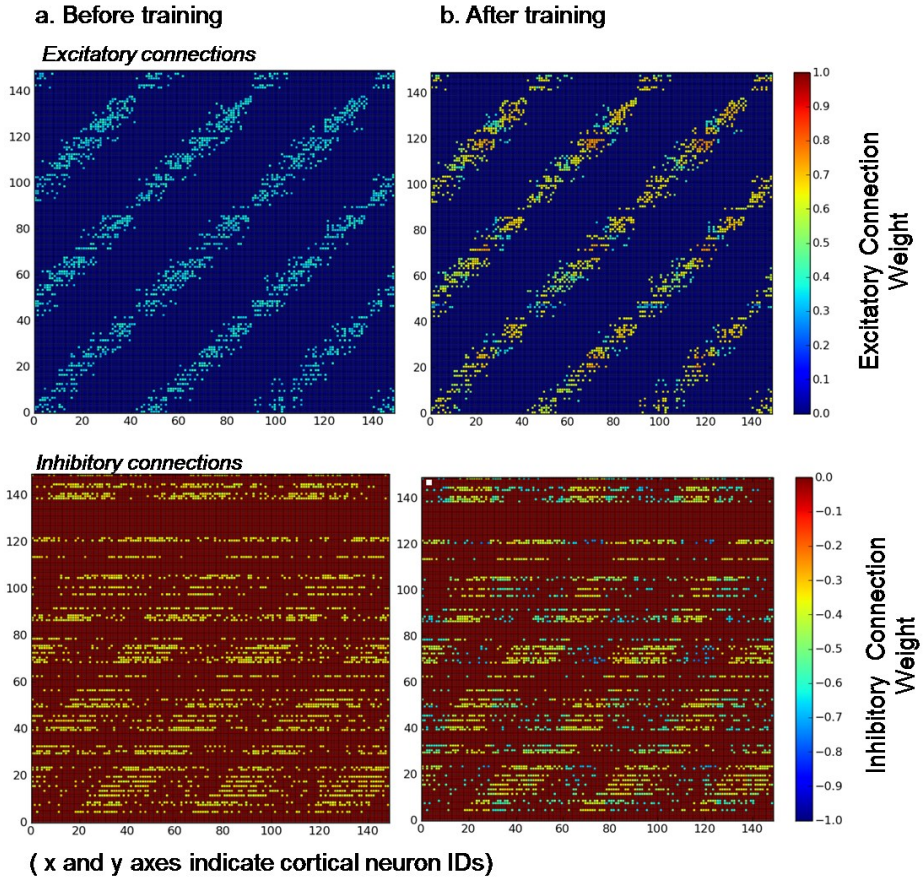


Figure 39 – Expt 2: Lateral Weights Before and After Training

The Spatiotemporal Response

Figure 40 shows a plot of the spatiotemporal response of the 48x48 output layer following the presentation of two dissimilar patterns before and after training. Initially, the network is nearly equally responsive to both patterns with the majority of the neurons firing at about 9ms (which is the integration time). After training about half of the neurons are silent and a spatially distinct patch of neurons respond to each of the patterns and the range of firing times of the responding populations are also different to the initial state. For example, in the response to pattern East the cortical neurons fire between 9 and 14ms whilst for pattern South they fire between 9 and 10ms.

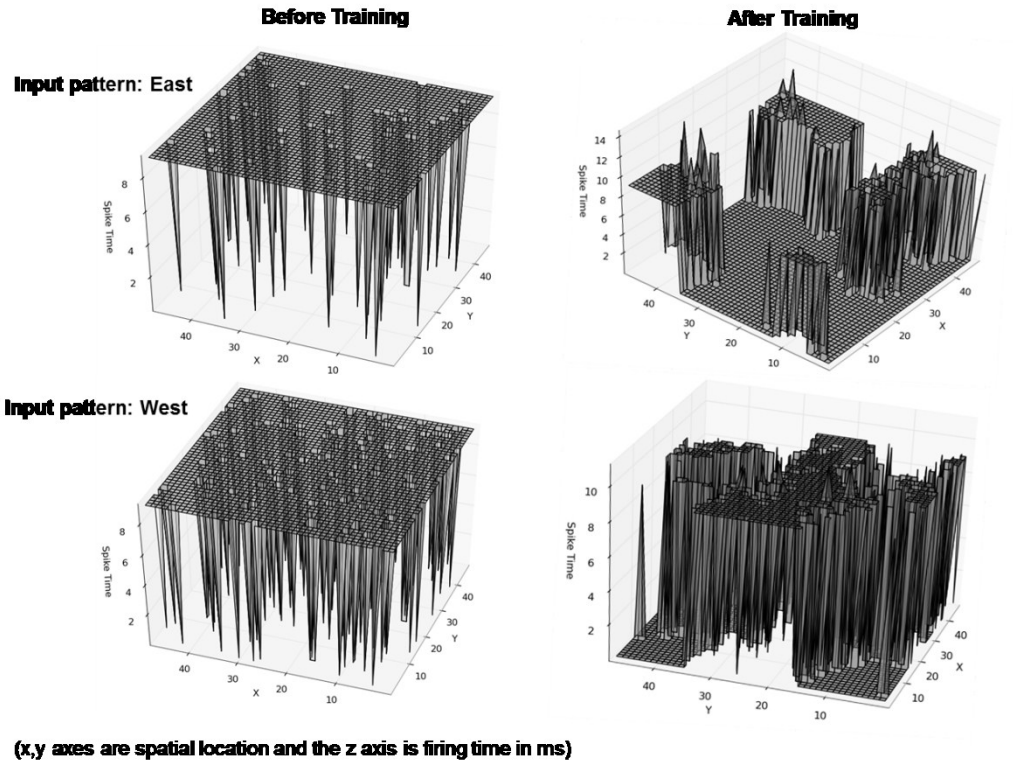


Figure 40 – Expt 2: Cortical layer response to two different patterns before and after training

Final Map Topography

The situation is qualitatively the same as for the 16x16 map, except there are more neurons available to represent each pattern. Figure 41shows the composite response of

the motor map before and after training. In Figure 41a, similarly to the smaller scale map, the responses to all the patterns overlap considerably and there is no indication of spatial organisation.

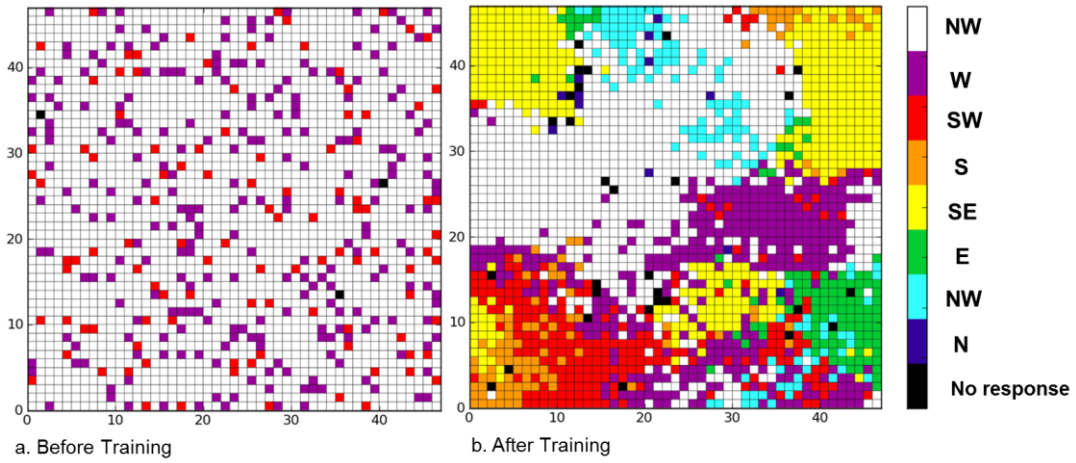


Figure 41– Expt 2: 48x48 Motor Map Topography

In Figure 41b, over 98% of the cortical neurons have developed a preference for at least one of the 8 directions during training (although there is still some overlap in the response) and neurons with the same preference are mainly grouped in the same spatial location(s). In the main, neighbouring patterns are also located near to each other, for instance patterns S (orange), SW (red) and W (purple) are next to each other as are patterns NW (cyan), E (green) and SE (yellow). At this scale, the map shows characteristics similar to the real directional map shown in Chapter 2, Figure 2.4.

8.4 Experiment 3 - *Motor Map Learning in the Humanoid Simulation*

Experimental setup

The same parameters as for the benchmark Experiment 1 (Section 8.2) were used except that the number of training cycles was reduced due to the extra time taken to run under the humanoid physics simulation. See Table 8 for a summary.

Parameter	Description	Value
Ap	LTP rate	0.1
Am	LTD rate	-0.105*Ap
N	Number of training cycles	5
P	Total number of patterns	800
η	Initial learning rate	0.5
η_r	Learning rate reduction	0.949

Table 8 - Parameters for Experiment 3

Main Results

In order to confirm that the motor map development process is not affected by running under the robot simulation, plots were produced similar to those of Figures 36 and 37. Figure 42 shows that the spatiotemporal response has qualitatively similar characteristics to those in Experiment 1.

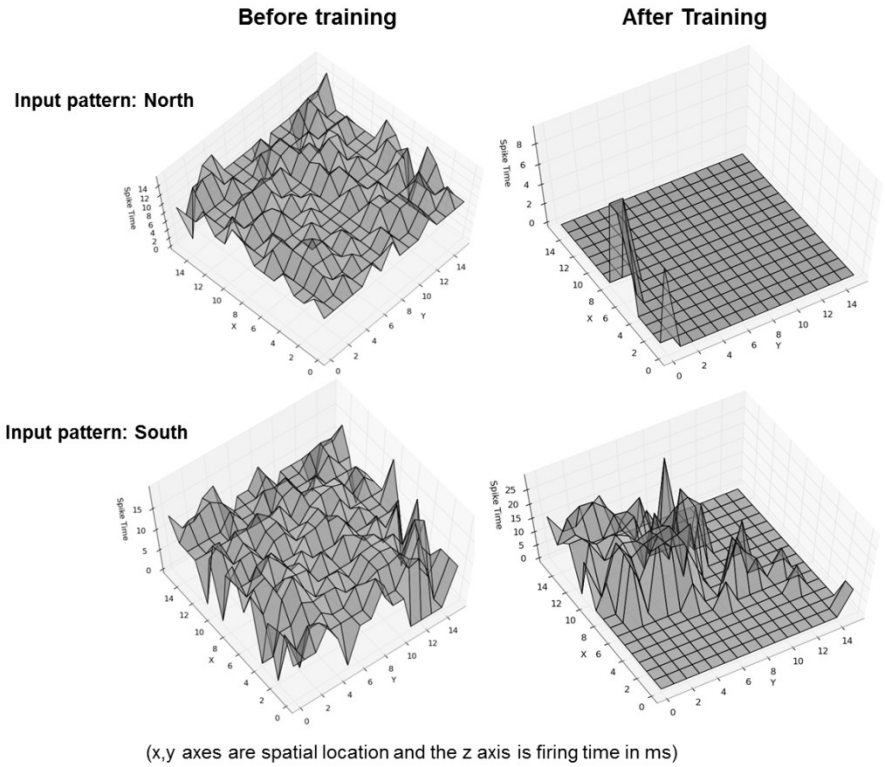


Figure 42 – Expt 3: Cortical layer response to two different patterns before and after training

Initially the response is spread over the whole of the map for both patterns and the range of firing times is within the same interval. After training both the spatial extents of the

responses has reduced and the range of firing times has changed. Figure 43shows the motor map before and after training. In Figure 43a, the results before training are very similar to those in Experiment 1: the responses overlap and there is no distinct spatial organisation. Figure 43b shows that after training the map is spatially organised with the responses to particular patterns grouped in patches.

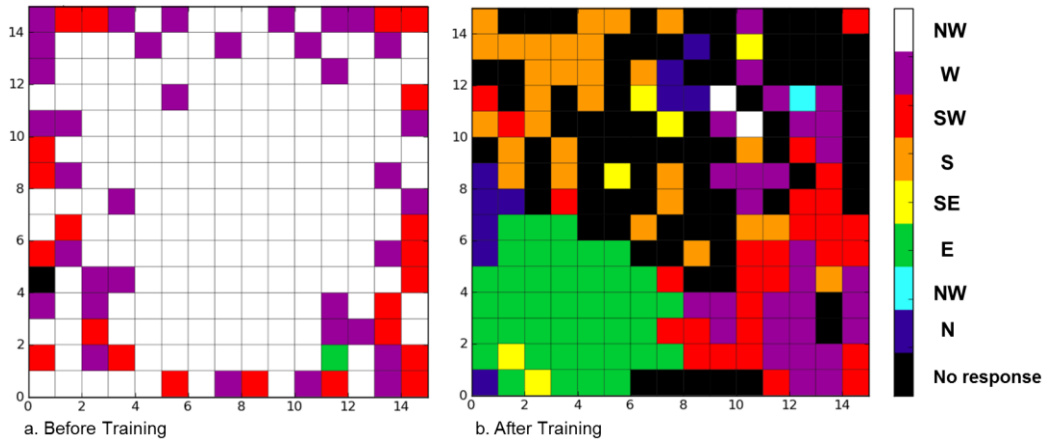


Figure 43 – Expt 3: 16x16 Motor Map Topography (from Humanoid Simulation)

Quantification of the Spatiotemporal Response

From the results so far, it has been established that there has been the appropriate kind of learning on both the afferent and lateral weights, that a topographic map representing the 8 directions is formed and that the cortical responses are at least visually distinct. However, do the spatiotemporal responses really identify the patterns distinctly? Chapter 6, Section 6.3 proposed an application of the van Rossum metric (van Rossum, 2001) to decode the motor map response in a quantitative way so that it could be used to implement sensorimotor integration and also testing with the humanoid simulation.
In order to verify this, a new 'test' dataset was created consisting of 160 patterns (using the same methods that were used to create the training datasets). The humanoid simulation was initialised using the map generated from the benchmark (Experiment 1, 16x16 motor map). The simulation was then run in a testing mode which does the following:

1. A pattern is read in from file

2. Neural processing is run to apply the pattern as input to the map

3. The map response is analysed using the van Rossum metric and a prediction of the pattern number is generated

4. The pattern number is converted to a turn direction (Left or Right) and a number of turns to execute

5. The movement sequence is initiated

6. Data on the actual pattern presented, the van Rossum scores, the neural prediction and the movement executed is collected

It was found that there was approximately 13% error between the actual presented pattern and the pattern predicted from the van Rossum metric scores, with the most common fault being pattern S classified as SW. There was no error in movement execution.

To verify how the system would cope with ambiguous inputs a small dataset was created with patterns deliberately midway between exemplars – i.e. 1/2, 3/4, 5/6, 7/8. Ideally the system should make a reasonable decision by choosing the best match. Running the testing process showed that the system does make reasonable decisions. In all cases the van Rossum metric scores are low for both exemplars and the system picks the lowest score, as expected.

8.5 Experiment 4 - Demonstration of Visual Directional Selectivity

Experimental setup

The setup as described in Chapter 4 was used. Therefore the learning rules incorporated a traditional learning rate parameter reduction. Table 9 gives a summary of the relevant experimental parameters. The purpose of this experiment was to replicate part of the work in Wenisch et al. (2005) which showed how the lateral weight changes developed

in response to repeated presentations of a 'preferred' pattern due to the action of the asymmetric STDP learning rule. The training phase consisted of presenting 20 instances of pattern 'East' keeping the learning rate at 1.0.

Parameter	Description	Value
A_{pa},	Afferent LTP rate	0.005
A_{ma}	Afferent LTD rate	-0.105* A_{pa}
A_{pl}	Lateral LTP rate	0.0005
A_{ml}	Lateral LTD rate	-0.105* A_{pl}
P	Total number of patterns	20
η	Learning rate	1.0

Table 9 - Parameters for Experiment 4

Characteristics of afferent and lateral weight training

Wenisch et al (2005) demonstrated that directional selectivity had developed by showing an extract of the lateral weight matrix before and after learning indicating how asymmetric strengthening had occurred on the side of the preferred direction. Here both afferent and lateral weights are examined for evidence of directional selectivity. Figure 44 shows a composite plot of the difference in the afferent weight matrix before and after training, calculated as (after-before) to show weight increase as positive and weight decrease as negative. The weight changes show a definite West to East pattern of strengthening on the preferred direction (West → East) and weakening on the null direction (East →West). Figure 45 shows the same type of plot for 4 cortical neurons and their lateral (excitatory) connections. Whilst the W→ E and E→W asymmetry is not as strong as for the afferent weights there is evidence for asymmetric strengthening/weakening in broadly these directions (in particular Figs 45a and c). It is likely that Wenisch et al (2005) were able to show this more clearly in their work as the input was a solid bar spanning the map extent and in this work the input is individual spike events from the DVS camera. Furthermore Wenisch et al. did not include afferent connections in their work, but instead fed the inputs directly into the cortical layer.

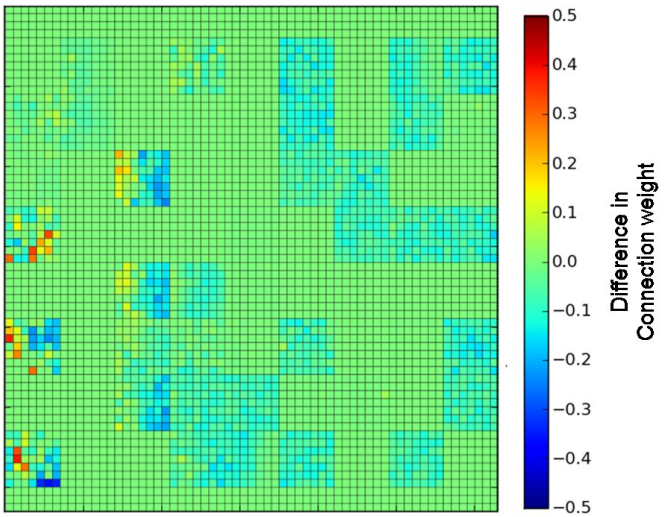


Figure 44 – Expt 4: Difference in afferent weights after training

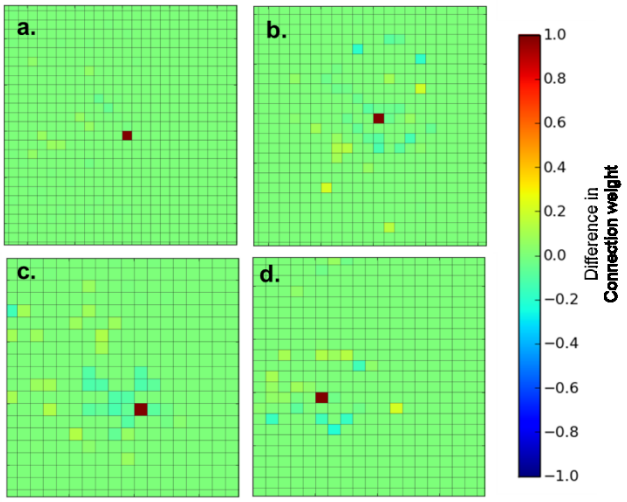


Figure 45 – Expt 4: Difference in lateral weights after training

Response to Null/Preferred directions

In order to test the overall response of the map to the null/preferred directions, learning was disabled and 15 instances each of pattern East (preferred direction) and pattern West (null direction) were presented to the trained and untrained versions of the maps. Table 10 shows the average of the map responses to the null and preferred patterns before and after training.

Pattern	Direction	Response (total spikes produced)
Before Training		
East	W → E	1404000
West	E→W	2012498
After Training		
East	W → E	1406010
West	E→W	776713

Table 10 – Expt 4: Responses to Null and Preferred Directions

This is 0.14% increase in response for the preferred pattern and 61% decrease for the null pattern. This clearly shows the existence of directional selectivity which is operating through inhibition of the null direction rather than enhancement of the preferred direction. As Wenisch et al. did not perform this analysis in their work it is not possible to compare these results with theirs.

8.6 Experiment 5 - *The Visual Map Benchmark*

Experimental setup

The setup as described in Chapter 4 was used. Therefore the learning rules incorporated a traditional learning rate parameter and learning rate reduction. Table 11 gives a summary of the experimental parameters.

Parameter	Description	Value
A_{pa},	Afferent LTP rate	0.005
A_{ma}	Afferent LTD rate	-0.105* A_{pa}
A_{pl}	Lateral LTP rate	0.0005
A_{ml}	Lateral LTD rate	-0.105* A_{pl}
P	Total number of patterns	100
η	Initial learning rate	1.0
η_r	Learning rate reduction	0.995

Table 11 - Parameters for Experiment 5

Training consisted of presenting patterns randomly from the set of 8 exemplar sequences recorded from the DVS 128 camera. Having looked closely at the directional

selectivity training as regards afferent and lateral weight changes in Experiment 4, this section concentrates on overall map development and the characteristics of the spatiotemporal response to individual patterns.

The Spatiotemporal Response

In analysing the spatiotemporal response for the visual map, as there is considerably more activity than in the case of the motor map, only neurons which have fired more than 20 times are selected and only their last firing time is plotted. Figure 46 shows the response to two patterns (North and South) before and after training. Here the characteristics of the spatiotemporal response are very different to those seen for the motor map due to the fact that there is now a strong temporal component already present in the input patterns. In fact it was discovered that there is actually a distinct spatial and temporal response to each pattern even before training: the left hand side of Figure 46 shows that many neurons fire in a S→N sequence for pattern North and the opposite direction for pattern South. In the right hand side of Figure 46 the characteristics of the responses after training are similar except that the overall response is reduced and there is less noise: there is a clearer sequence of activation over time in the preferred direction. Experiment 4 showed that the direction selectivity arises mainly from inhibition on the null direction and this fits with these results.

Final Map Topography

In analysing the topographic visual map only neurons which have fired more than 20 times are plotted. Figure 47 shows the composite response of the visual map before and after training. In Figure 47a, the case for the map before training it can be seen that there are actually features of a directionally selective map in the initial state (for example compare the experimental map shown in Chapter 2, Figure 2.4) which is somewhat surprising considering previous modelling works (to the author's knowledge) have not shown this to be the case. However the map responses before training do have

a considerable overlap. In Figure 47b after training, the map is still directionally selective in that it represents all 8 patterns, but as the activity has been reduced the response is much sparser.

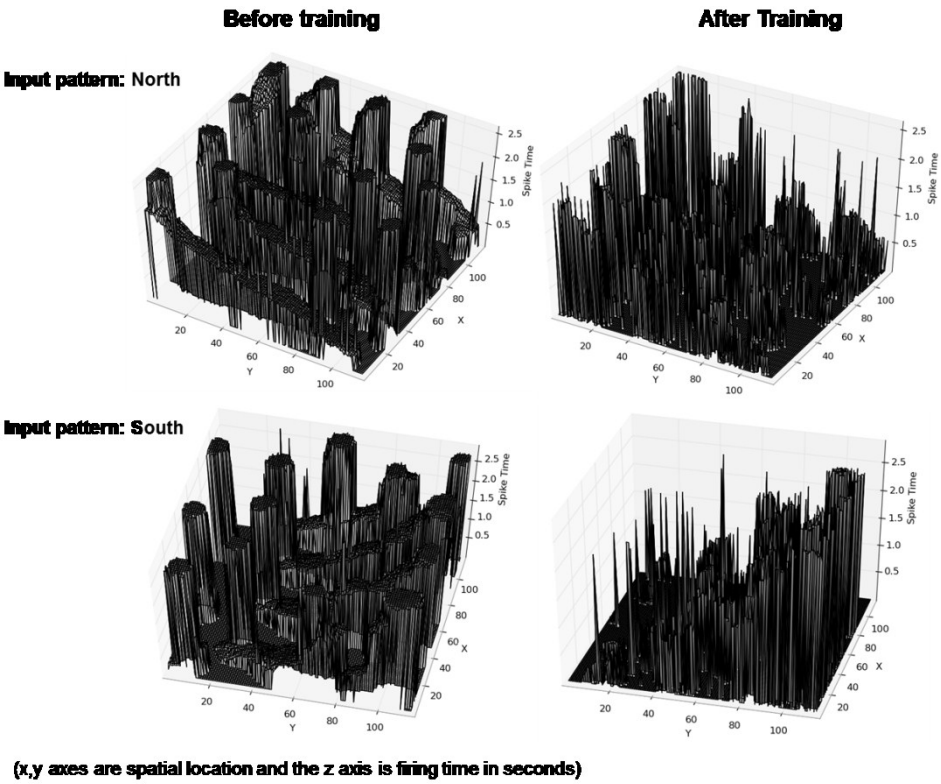


Figure 46 – Expt 5: Cortical layer response to two different patterns before and after training

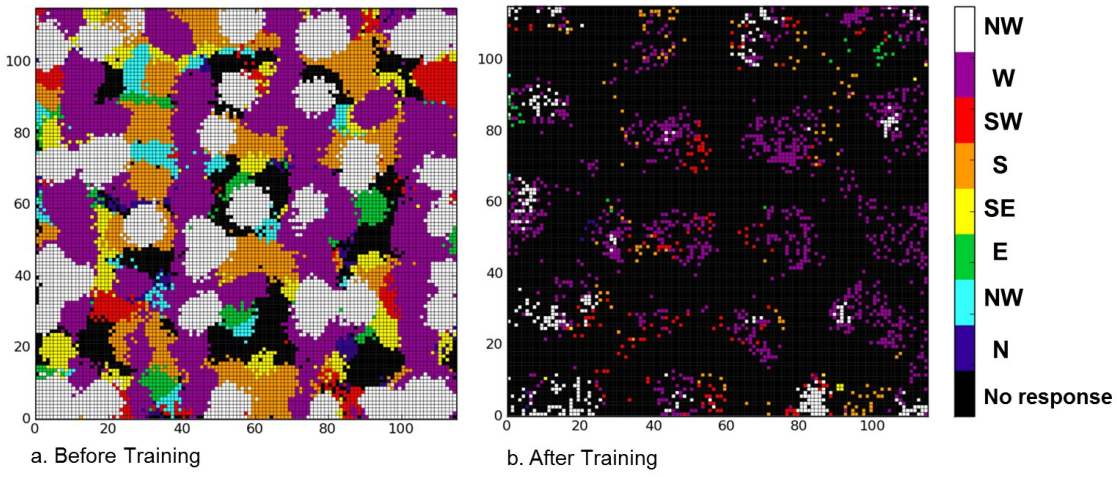


Figure 47 – Expt 5: Visual Map Topography

Quantification of the spatiotemporal response

The previous subsection showed that, at least by visual inspection, there appeared to be a selective response to the 8 exemplar patterns both before and after training. However, as with the motor map analysis it was necessary to do a quantitative analysis to see how well the map responses actually identified the patterns. In view of the unexpected result of a directionally selective response before training it was of interest to see how selective this was also. The van Rossum metric analysis was done comparing the response to each pattern to the response for all other patterns for both the before and after maps. The scores between each pair of patterns are given in Table 12. Note that in this analysis (and the training) perturbed versions of the inputs are not used, therefore a 'match' is perfect, i.e. a score of 0. The non-match scores are around 0.6 which is the same as for the motor map analysis. In the top half of Table 12, the scores for the 'after training' case show that the perfect matches are between patterns and themselves. There is also some variation in scores indicating that similar patterns have lower scores. For example, in the row for NE, the second lowest score (of 0.4908) is with its neighbouring pattern, E. In the bottom half of Table 12, the scores for 'before training' show that apart from the perfect matches between patterns and themselves, all of the other scores are high and do not indicate similarity between neighbouring patterns at all. In a more applied scenario, it would be necessary to train with a better input pattern repertoire – i.e. many different instances of the same pattern (created by multiple recordings of the same direction with the DVS camera). In this case there would be much more ambiguity about the identity of patterns and it is likely that the results of a van Rossum analysis would show that the classification is more flexible after training.

After Training								
	N	NE	E	SE	S	SW	W	NW
N	0	0.5219	0.5514	0.5426	0.6117	0.5369	0.5349	0.5036
NE	-	0	0.4908	0.5708	0.6189	0.5774	0.5892	0.5485
E	-	-	0	0.5928	0.5832	0.5123	0.5520	0.5161
SE	-	-	-	0	0.5357	0.4503	0.5030	0.5054
S	-	-	-	-	0	0.5265	0.5772	0.5762
SW	-	-	-	-	-	0	0.4574	0.4752
W	-	-	-	-	-	-	0	0.5056
NW	-	-	-	-	-	-	-	0
Before Training								
	N	NE	E	SE	S	SW	W	NW
N	0	0.5996	0.5928	0.5914	0.6172	0.5925	0.6039	0.6044
NE	-	0	0.5691	0.5965	0.6323	0.5887	0.6027	0.5875
E	-	-	0	0.6260	0.6261	0.5831	0.5719	0.5835
SE	-	-	-	0	0.5982	0.5890	0.6091	0.6094
S	-	-	-	-	0	0.5875	0.6173	0.6236
SW	-	-	-	-	-	0	0.5712	0.5704
W	-	-	-	-	-	-	0	0.5901
NW	-	-	-	-	-	-	-	0

Table 12 – Van Rossum Metric Analysis for Visual Map Responses.

9 Training with Adaptive Plasticity

9.1 Overview

This chapter presents the results of training the motor and visual maps using the Adaptive Plasticity methods described in Chapter 5. Section 9.2 firstly describes the results of incorporating the Plasticity Resource to the motor map 'benchmark' (16x16 cortical layer) to replace the conventional learning rate parameter. The purpose of this experiment was to confirm that qualitatively the same results are achieved but with the added benefit of autonomous control of training. Secondly, an experiment is done with a new set of data to show how the Adaptive Plasticity methods can be used to control training when the data has not been seen before and so the quantity and composition of the training data required is unknown. Section 9.3 confirms that the Adaptive Plasticity methods work for the scaled up motor map (48x48 cortical layer) and, for completeness, Section 9.4 confirms that using Adaptive Plasticity within the humanoid training simulation again gives similar results.

Section 9.5 describes the results of incorporating the Plasticity Resource to the visual map 'benchmark' to replace the conventional learning rate parameter. As for the motor map, the purpose of this experiment was to confirm that qualitatively the same results are achieved but with the added benefit of autonomous control of training. Sections 9.6 and 9.7 describe the results of applying the rewiring methodology described in Chapter 5 to the motor (16x16 cortical layer) and visual maps to establish whether the same map results can be achieved with a sparser final connectivity. Full discussion of the results is left until Chapter 11.

9.2 Experiment 6 – The Motor Benchmark with Adaptive Plasticity

Experiment 6a - Experimental Setup

This experiment was performed incorporating the Plasticity Resource (PR) with modifications to the motor learning rules as described in Chapter 5, Section 5.2 but with the same predefined datasets as used in the original benchmark. Table 13 gives a summary of the relevant experimental parameters.

Parameter	Description	Value
Ap	LTP rate	0.02
Am	LTD rate	-0.105*Ap
N	Number of training cycles	20
P	Total number of patterns	3200

Table 13 - Parameters for Experiment 6a

In this experiment the rate of map training was controlled solely by the plasticity resource. As the scaling to a maximum synapse weight had been removed from the learning equations a lower STDP learning rate was used to ensure that the final maximum excitatory and inhibitory weights did not end up too large. Due to the action of the constantly decreasing plasticity resource twice as many training cycles were required compared to the benchmark.

Experiment 6a - Main Results

In order to confirm whether the training setup qualitatively matched the results of the benchmark experiment, plots of the output response for patterns North and South were produced for comparison with those in Chapter 8, Figure 36. These are shown in Figure 48. It can be seen that there is indeed similar behaviour. Before training the responses are difficult to distinguish as there is activity from a large proportion of the neurons. After training the responses are spatially distinct. Figure 49 shows a graph of the plasticity resource (PR) value as measured at the end of each of the 20 training cycles. This confirms that the PR trace behaves as expected: an initial high value which

decreases gradually as the network learns the range of input patterns. In the final few cycles the trace levels off and changes remain around 0.01 which indicates when training is complete.

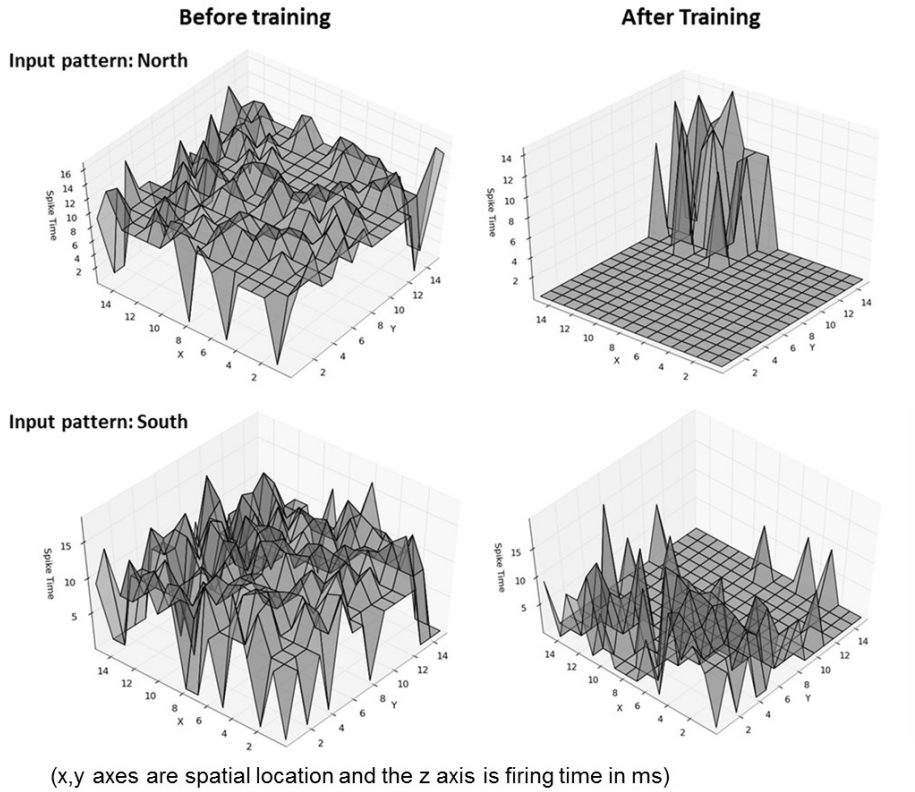


Figure 48 – Expt 6a: Cortical layer response to two different patterns before and after training

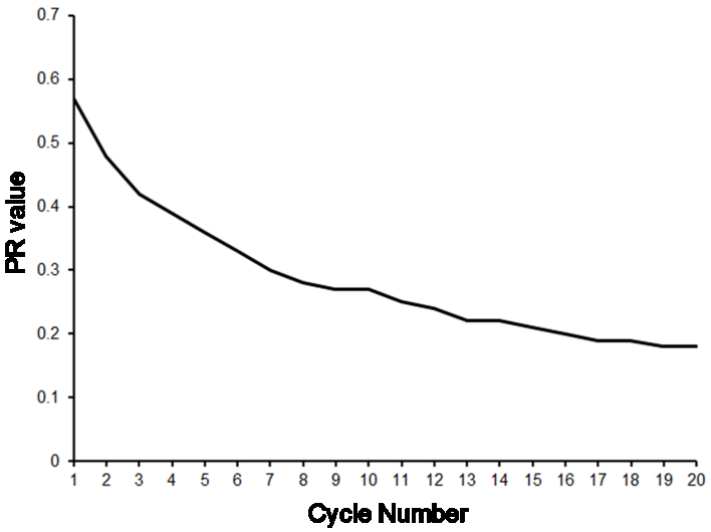


Figure 49 – Expt 6a: Plasticity Resource trace

Experiment 6b - Experimental Setup

Up to this point the same set of directional input patterns had been used for all motor map experiments and presented in the form of predefined datasets with a fixed number and range of input patterns. In order to see the real benefit of using the PR trace to control learning it was necessary to create some new patterns where the amount of training required was unknown. A set of four new exemplar patterns were created from data generated from the humanoid robot simulation previously described in Chapter 6. The patterns consisted of 16 values taken from joint angles required to make specific robot poses: left and right leg kick and left and right arm point. The values were converted into spike times using the Linear Temporal Encoding method described in Maass (1997). The patterns naturally contained a mixture of noise and salient data as not all joints are involved in all poses. More details of the creation of these input patterns are given in Appendix A. The training setup was also changed to select a training pattern randomly from the four exemplars and perturb the values thus simulating a sequence of random robot movements. Training was run in blocks of 160 patterns purely for the purpose of capturing the state of the network during the course of training, but there was no plan to run a specific amount of patterns. Apart from these changes to the input, the other experimental parameters were the same as for Experiment 6a.

Experiment 6b - Main Results

In this experiment the training duration was guided entirely by the PR trace, which is shown in Figure 50. Training was stopped after 2080 patterns, when the PR trace value changed by only 0.01 units. Figure 51 shows the network response to 'Right Kick' and 'Right Arm Point' patterns before and after training. As seen in previous experiments, the before case (left hand side of plot) shows no significant distinction between the patterns: the majority of neurons are firing together at about the same times. The final

response (right hand side) is very much sparser and a spatiotemporally distinct response for the new patterns has developed. After training, the majority of neurons do not respond and both the spatial range and firing times of the responding neurons has become more distinctive for each pattern.

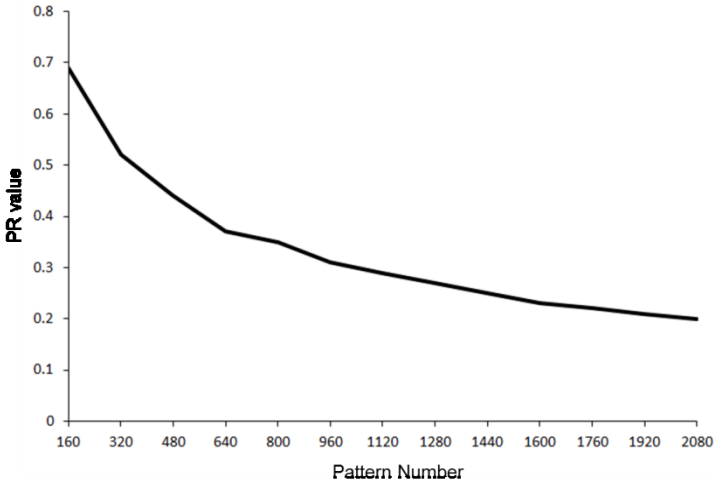
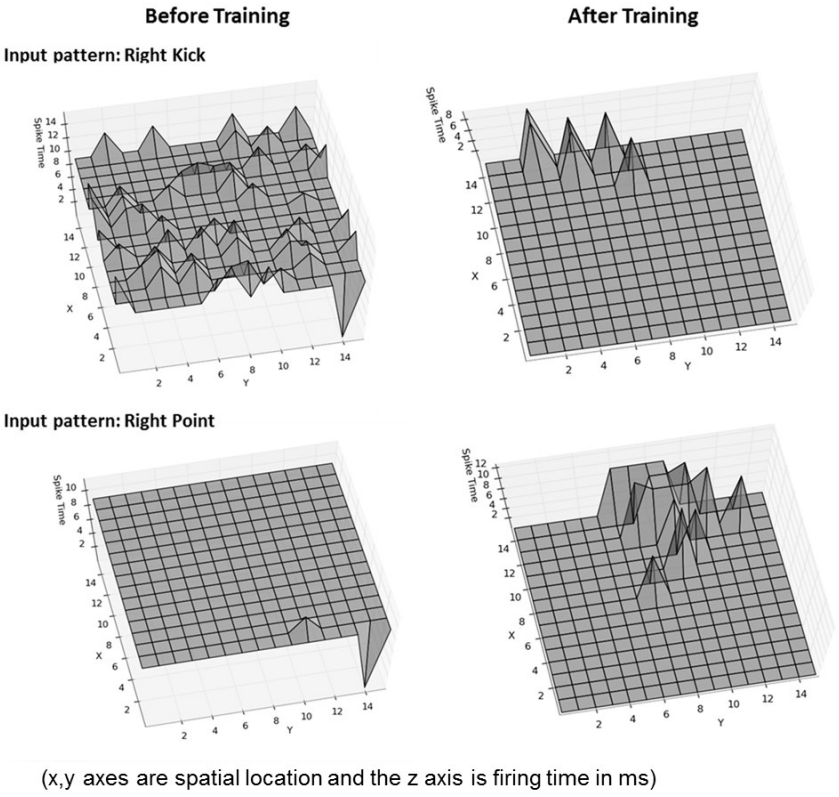


Figure 50 – Expt 6b: Plasticity Resource trace



(x,y axes are spatial location and the z axis is firing time in ms)

Figure 51 – Expt 6b: Cortical layer response to two different patterns before and after training

9.3 Experiment 7 - The Scaled Up Motor Map with Adaptive Plasticity

Experiment 7- Experimental Setup

The same setup as given in Table 6 for experiment 6a was used, except that patterns were presented randomly online until the PR trace indicated that training had completed.

Experiment 7- Main Results

Training was stopped after 500 patterns when the PR trace value changed by only 0.01 units. The PR trace is shown in Figure 52 and exhibits the same pattern of reduction as previous experiments in this chapter.

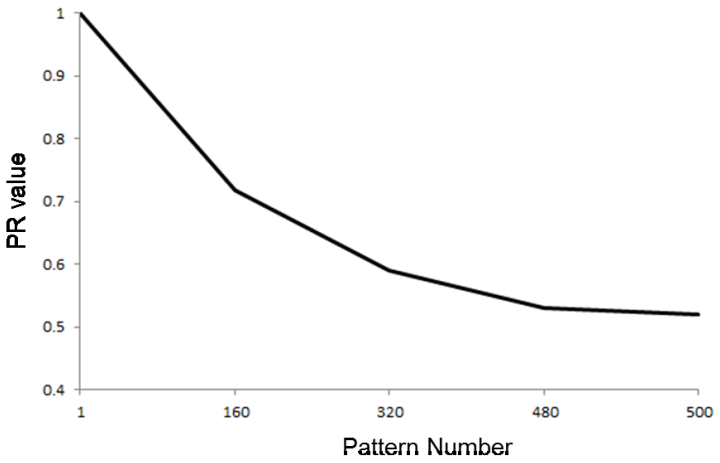


Figure 52 – Expt 7: Plasticity Resource trace

In terms of the map response to different patterns, Figure 53 shows the network response to 'East' and 'West' patterns before and after training. Comparing this to the experiment without Adaptive Plasticity (Chapter 8, Fig 40), the results are qualitatively similar. Before training, both patterns elicit a response from the majority of the neurons across the area of the map around the integration time of 9ms. The final response is very much sparser and a spatiotemporally distinct response for the patterns has developed.

9.4 Experiment 8 - Adaptive Plasticity in the Humanoid Simulation

Experiment 8 - Experimental Setup

The same setup as experiment 6a was used except that the training patterns were presented randomly from the 8 directional exemplar patterns. Training was run in blocks of 160 patterns purely for the purpose of capturing the state of the network during the course of training but there was no plan to run a specific amount of patterns.

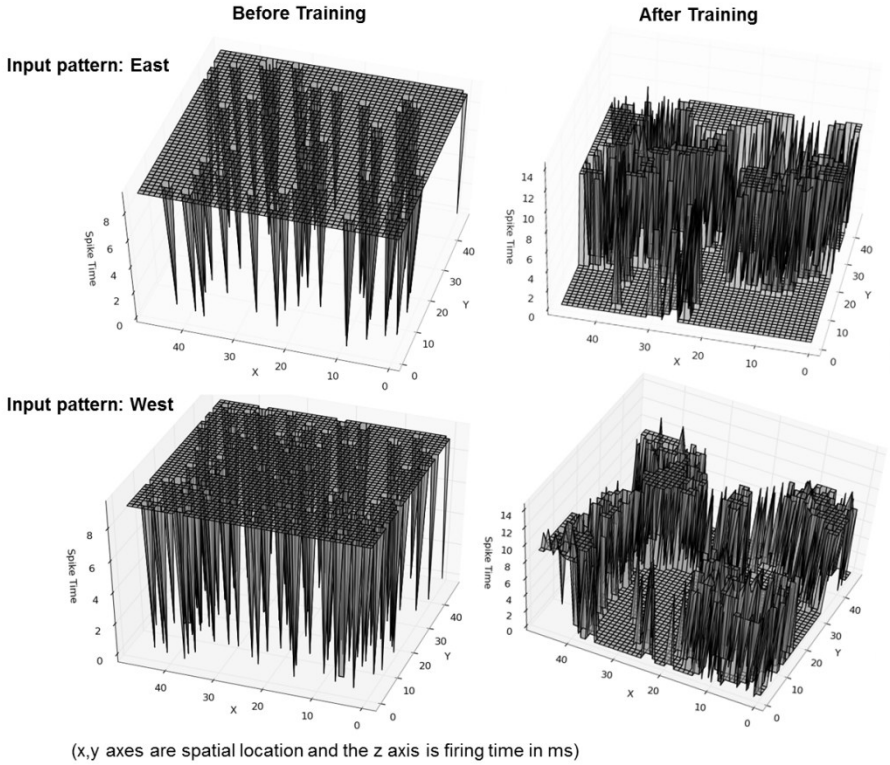


Figure 53 – Expt 7: Cortical layer response to two different patterns before and after training

Experiment 8 - Main Results

Training was stopped after 2080 patterns when the PR value changed by only 0.01 units. The PR trace is shown in Figure 54 and exhibits the same pattern of reduction as seen in previous experiments in this chapter. The output response for the final network for patterns North and South is shown in Figure 55. Comparing this to the same response from the network in the benchmark (Chapter 8, Figure 42) there is similar behaviour in that the response after training is spatiotemporally distinct from that of the initial

network and the response to the two dissimilar patterns is quite different.

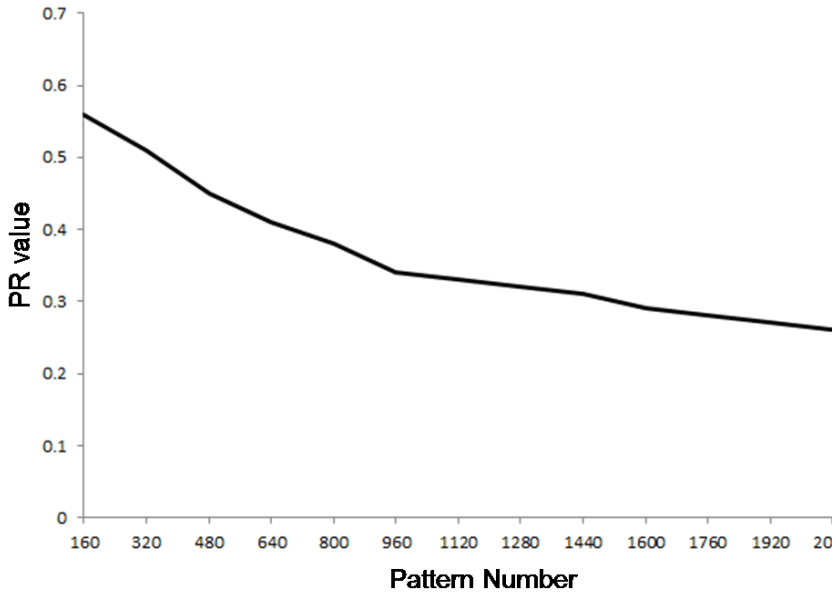


Figure 54 – Expt 8: Plasticity Resource trace

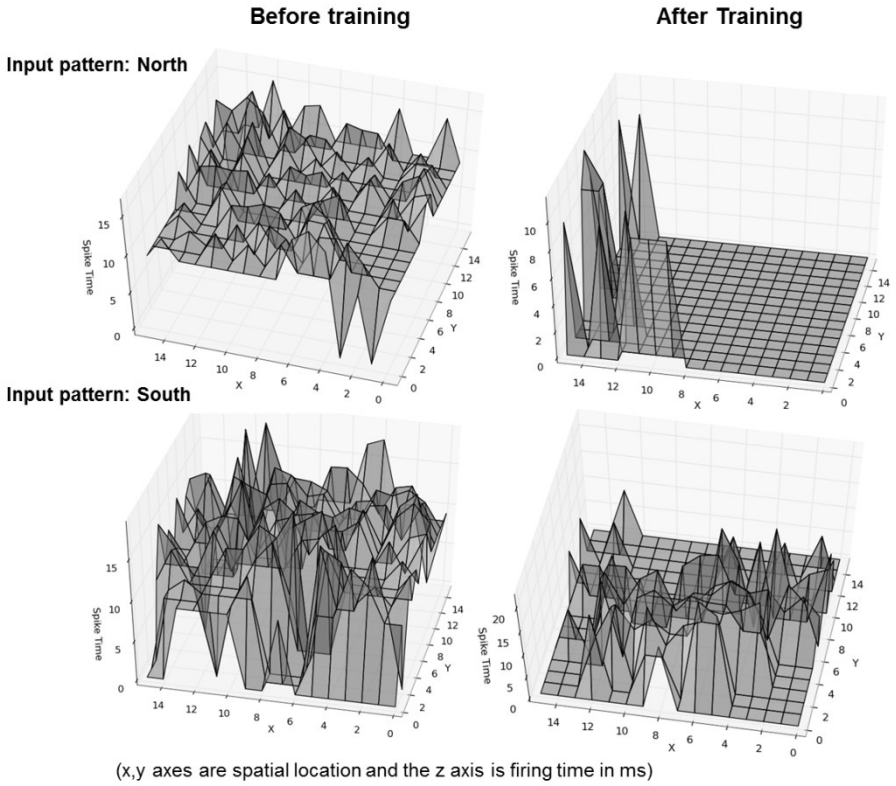


Figure 55 – Expt 8: Cortical layer response to two different patterns before and after training

9.5 Experiment 9 - The Visual Benchmark with Adaptive Plasticity

Experiment 9 - Experimental Setup

This experiment was performed incorporating the Plasticity Resource (PR) with modifications to the visual learning rules as described in Chapter 5, Section 5.2. Table 14 gives a summary of the relevant experimental parameters.

Parameter	Description	Value
A_{pa},	Afferent LTP rate	0.003
A_{ma}	Afferent LTD rate	-0.105* A_{pa}
A_{pl}	Lateral LTP rate	0.0003
A_{ml}	Lateral LTD rate	-0.105* A_{pl}

Table 14 - Parameters for Experiment 9

In this experiment the rate of map training was controlled solely by the plasticity resource. As the scaling to a maximum synapse weight had been removed from the learning equations lower STDP learning rates for both afferent and lateral learning were used to ensure that the final maximum weights did not end up too large. As before training consisted of presenting patterns randomly from the set of 8 exemplar sequences recorded from the DVS 128 camera.

Experiment 9 - Main Results

Training was stopped after 100 pattern presentations, as the PR trace value had changed by only 0.01 units. The results showed that the self-regulating learning regime can qualitatively reproduce the results of the visual benchmark experiment. The output response for the final network for patterns North and South is shown in Figure 56. Comparing this to the same response from the network in the benchmark without Adaptive Plasticity (Chapter 8, Figure 46) the results are qualitatively similar.

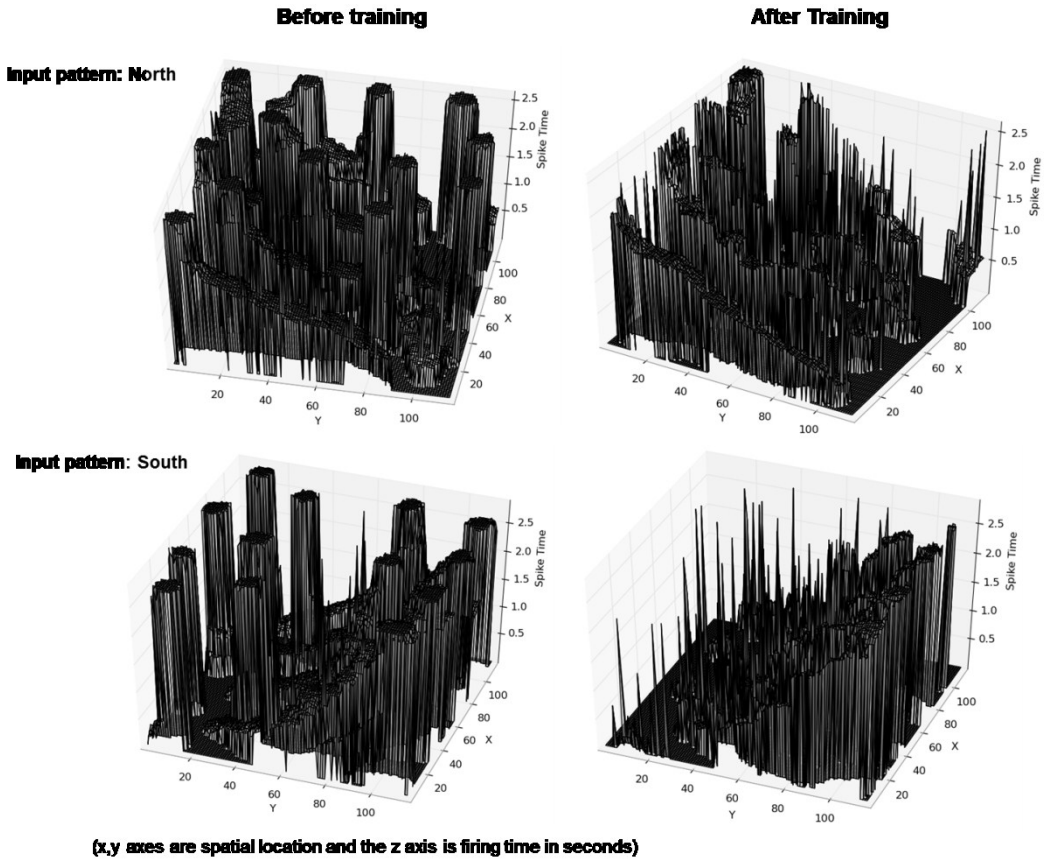


Figure 56 – Expt 9: Cortical layer response to two different patterns before and after training

As in the case without Adaptive Plasticity, there is a strong spatiotemporally distinct response before as well as after training. After training, the number of responders has reduced but the characteristics of the response are the same. Figure 57 shows a graph of the plasticity resource (PR) value as measured at the end of each of the training cycles. Here we see that the PR trace behaves in the same way as for previous experiments in this chapter.

9.6 Experiment 10 - The Motor Benchmark with Rewiring

Experiment 10 - Experimental Setup

This experiment was performed using the same setup as Experiment 6a (motor benchmark plus Adaptive Plasticity) with the addition of the rewiring regime described in Chapter 5, Section 5.3. Table 15 gives a summary of the parameters.

Parameter	Description	Value
Ap	LTP rate	0.02
Am	LTD rate	-0.105*Ap
MaxConnE	Max exc connections	30
MaxConnI	Max inh connections	120
PruneThresh	Pruning weight threshold	< 0.3 (exc) , > -0.3 (inh)
N	Number of training cycles	10
P	Total number of patterns	1600

Table 15 - Parameters for Experiment 10

As in the original benchmark, predefined datasets of motor input patterns were used.

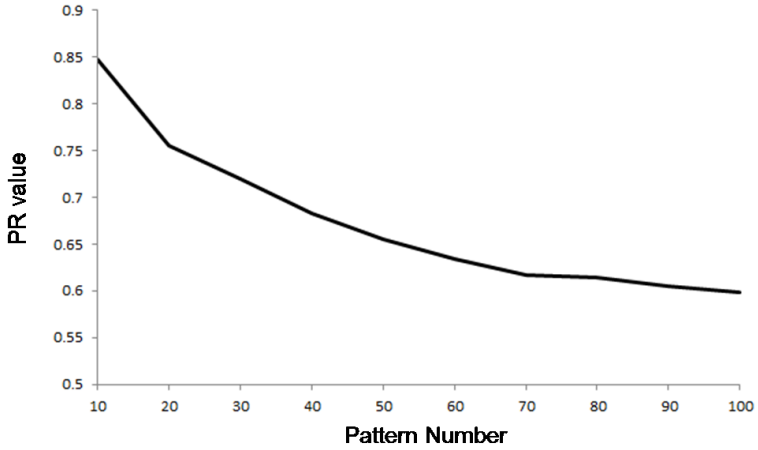


Figure 57 – Expt 9: Plasticity Resource trace

Experiment 10 - Main Results

The results showed that a learning regime with full Adaptive Plasticity (the Plasticity Resource to control training plus rewiring) can qualitatively reproduce the results of the motor benchmark experiment. The output response for the final network for patterns North and South is shown in Figure 58. Comparing this to the response from the network in the benchmark without Adaptive Plasticity (Chapter 8, Figure 36) the results are qualitatively similar: the response after training is spatiotemporally distinct from that of the initial network and the response to the two dissimilar patterns is quite different.

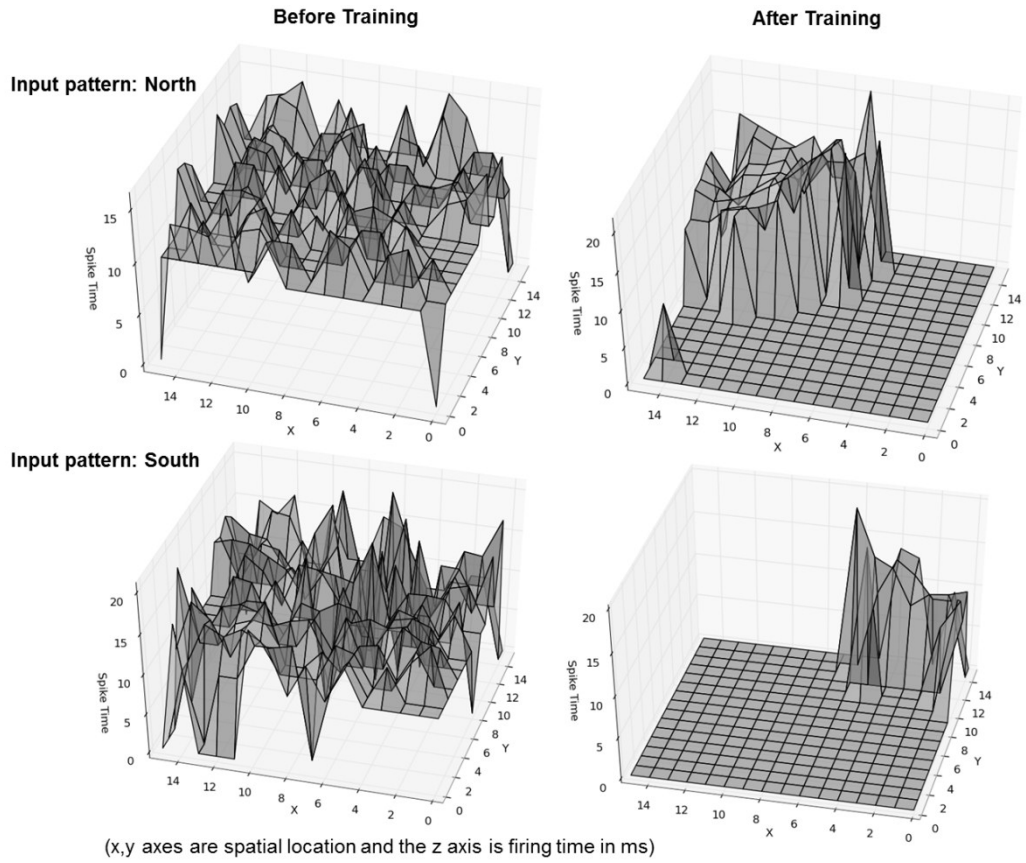


Figure 58 – Expt 10: Cortical layer response to two different patterns before and after training

Figure 59 shows a graph of the plasticity resource (PR) value as measured at the end of each of the 10 training cycles. Here we see that the PR trace behaves in the same way as for previous experiments in this chapter.

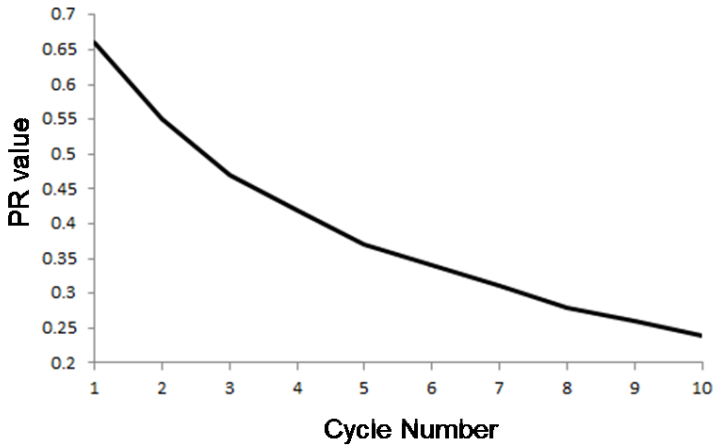


Figure 59 – Expt 10: Plasticity Resource trace

Experiment 10 – Rewiring Analysis

The main aim of adding a rewiring regime to the learning was to improve the adaptivity and efficiency of the map. So far it has been established that adding rewiring does not disrupt the results achieved from the benchmark motor map using the Plasticity Resource (PR) to control training. As part of experiment 10 the total number of connections (excitatory and inhibitory) created and pruned was collected after each training cycle. Figure 60 shows the total number of active connections per neuron before and after training.

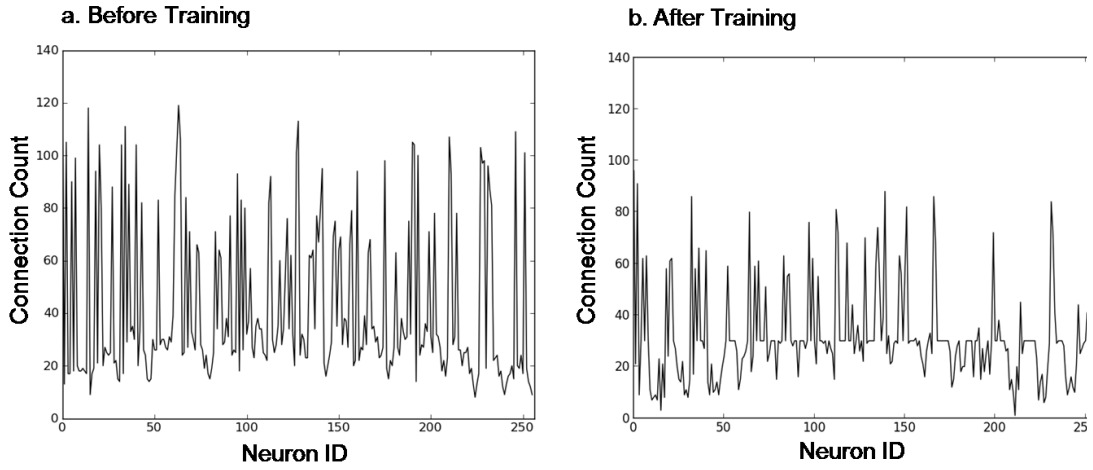


Figure 60 – Expt 10: Connection Counts per Neuron Before and After Training

Figure 60a shows the state of the connectivity in the initial network for all cortical neurons. The minimum values are around 30 connections (excitatory neurons) and the maximum around 120 connections (inhibitory neurons). Figure 60b shows that after training, the maximum count is a lot lower (around 90) and the minimum count is also a lot lower (less than 10). Table 16 shows some summary rewiring statistics. In terms of overall connectivity, the total number of connections is reduced by over 30%. It is also notable that the amount of pruning and creation ongoing during is high: many connections are pruned/created repeatedly but the core connections which maintain the response are obviously being kept as the results are qualitatively similar to the benchmark without rewiring.

Starting connection count	11597
Finishing connection count	7779
Total pruned over 10 cycles	41784
Total Created over 10 cycles	39008

Table 16 – Expt 10: Rewiring statistics

This indicates that using rewiring potentially increases the ability of the network to respond rapidly to changes in the input. It should be noted that it is considered that in real systems synaptogenesis and pruning operates on a different (slower) timescale than weight plasticity (Chklovskii et al., 2004). Nothing has specifically been done in this model to enforce a difference in timescale and it seems that rewiring here is actually operating on at least as fast a timescale as the normal plasticity.

9.7 Experiment 11 - The Visual Benchmark with Rewiring

Experimental setup

This experiment was performed using the same setup as Experiment 9 (visual benchmark plus Adaptive Plasticity) with the addition of the rewiring regime described in Chapter 5, Section 5.3. Table 17 gives a summary of the parameters.

Parameter	Description	Value
A_{pa},	Afferent LTP rate	0.003
A_{ma}	Afferent LTD rate	-0.105* A_{pa}
A_{pl}	Lateral LTP rate	0.0003
A_{ml}	Lateral LTD rate	-0.105* A_{pl}
MaxConnE	Max exc connections	30
MaxConnI	Max inh connections	800
PruneThresh	Pruning weight threshold	< 0.3 (exc) , > -0.3 (inh)

Table 17 - Parameters for Experiment 11

As in the original benchmark, patterns were presented randomly from the set of 8 exemplar inputs pre-recorded from the DVS 128 camera.

Experiment 11 - Main Results

The results showed that a learning regime with full Adaptive Plasticity (the Plasticity Resource to control training plus rewiring) can qualitatively reproduce the results of the visual benchmark experiment. The output response for the final network for patterns North and South is shown in Figure 61. Comparing this to the response from the network in the benchmark without Adaptive Plasticity (Chapter 8, Figure 46) the results are qualitatively similar. The initial state of the network shows a distinct spatiotemporal response which is refined by training to give a sparser, clearer sequence of activation over time in the preferred direction.

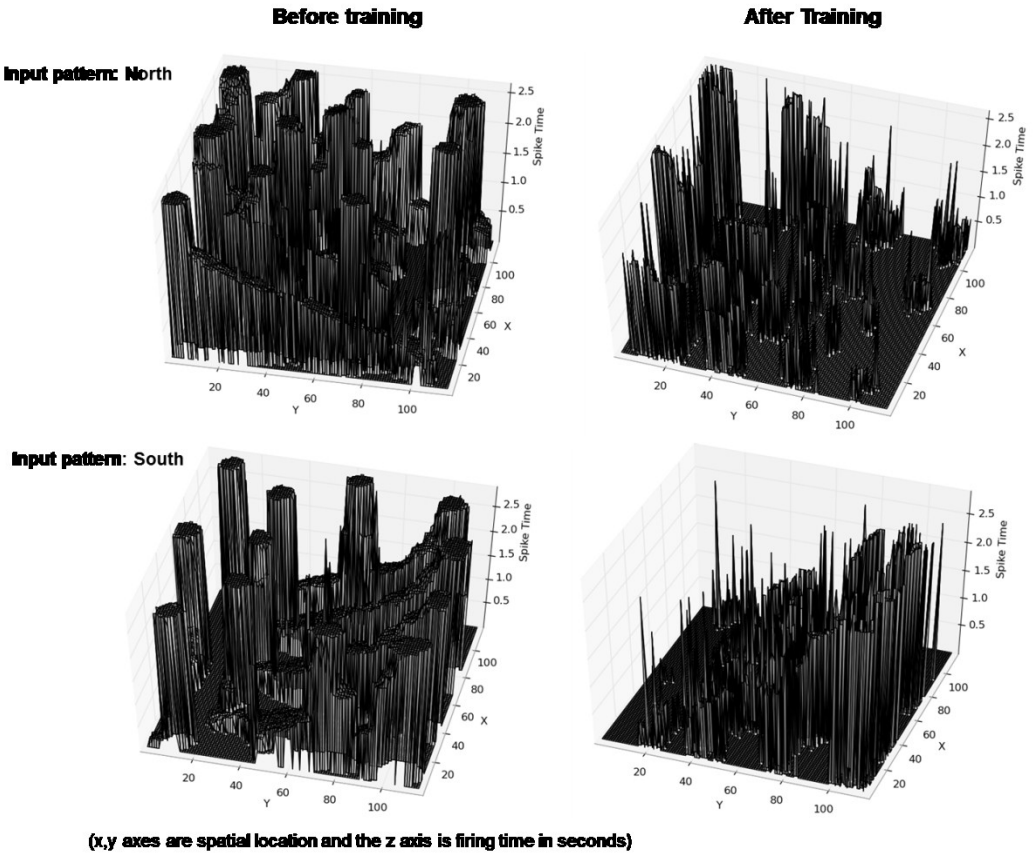


Figure 61 – Expt 11: Cortical layer response to two different patterns before and after training

Figure 62 shows a graph of the plasticity resource (PR) value as measured at the end of each of the 10 training cycles. Here we see that the PR trace behaves in the same way as for previous experiments in this chapter.

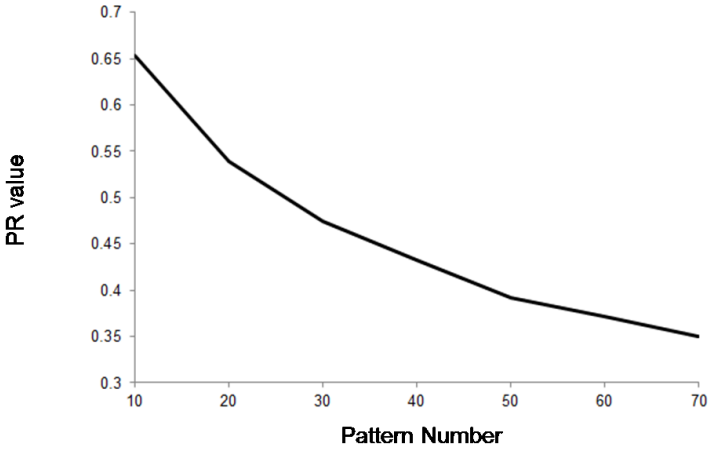


Figure 62 – Expt 11: Plasticity Resource trace

Experiment 11 – Rewiring Analysis

From the plots in the previous sections, it has been established that adding rewiring does not disrupt the results achieved from the benchmark visual map using the Plasticity Resource (PR) to control training. As part of experiment 11 the total number of connections (excitatory and inhibitory) created and pruned was collected after each training cycle. Figure 63 shows the total number of active connections per neuron before and after training. Figure 63a shows the state of the connectivity in the initial network for the first 1000 cortical neurons. The minimum values are around 30 connections (excitatory neurons) and the maximum values around 500 connections (inhibitory neurons). Figure 63b shows that, after training there is little different in the minimum and maximum values, and the changes are very small. Table 18 shows the rewiring statistics.

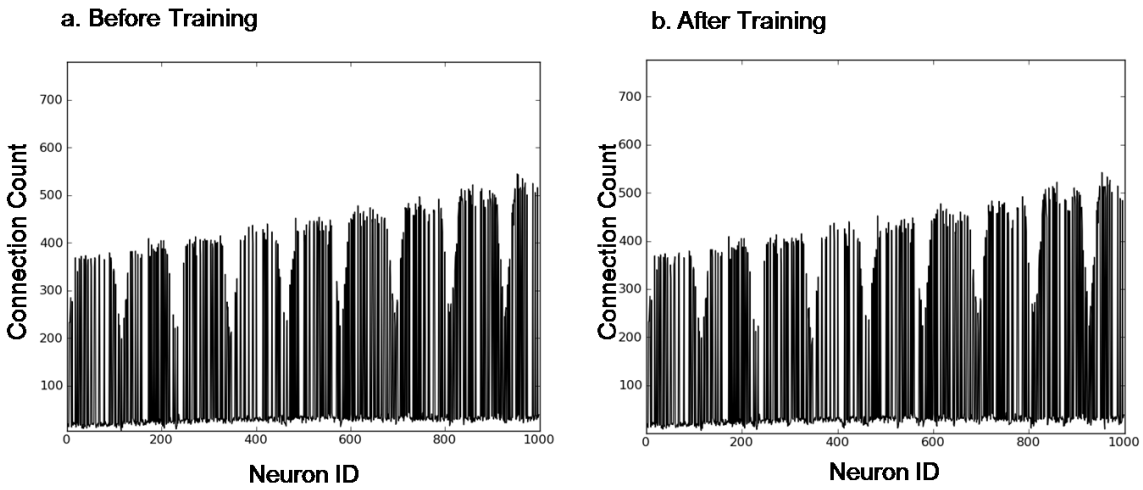


Figure 63 – Expt 11: Connection Counts per Neuron Before and After Training

Starting connection count	2737451
Finishing connection count	2701603
Total pruned over 10 cycles	35848
Total Created over 10 cycles	194452

Table 18 – Expt 11: Rewiring Statistics

In terms of overall connectivity, the total number of connections is reduced by only around 1%. The amount of pruning and creation going on during training is again high: many connections are pruned/created repeatedly but the core connections which maintain the response are obviously being kept as the results are qualitatively similar to the benchmark without rewiring.

10 Coupled Map Training

10.1 Overview

This chapter presents the results of coupling pre-existing motor and visual maps using the methods described in Chapter 7. Section 10.2 describes the main experiment coupling a 48x48 motor map with a 116x116 visual map. The motor map analysis of Chapter 8 is repeated for this 'new' motor map to discover how allowing visual input to control the motor map modulates the response, and whether a recognisable topographic map is still produced. As rewiring also plays an important part in the coupling some analysis is done on the balance of synapse pruning and creation and whether the final connectivity is reduced compared to the initial state. Full discussion of the results is left until Chapter 11.

10.2 Experiment 12 – Coupling the Motor and Visual Maps

Experiment 12 - Experimental setup

Table 19 gives a summary of the parameters used in this experiment

Parameter	Description	Value
Ap	LTP rate	0.1
Am	LTD rate	-0.105*Ap
MaxConn	Max connections	850
PruneThresh	Pruning weight threshold	< 0.4
P	Total number of patterns	50

Table 19 - Parameters for Experiment 12

Motor and visual input patterns were presented randomly according to the training process described in Chapter 7, Section 7.4.

Experiment 12 - Spatiotemporal response

To examine the way that the visual input alone affects the motor response via coupling, plasticity was disabled as well as the motor input. Recordings of the motor map

response to only input from the visual cortical layer were then taken using a map setup in an initial (untrained) and trained state. Figure 64 shows the spatiotemporal response of the motor map to the patterns East and West before and after training. The first thing to note is that the responses are not similar to the case of the pure motor map response (compare Chapter 8, Fig 40), in particular the time period over which the neurons fire has now expanded to fit the range of firing times of the visual neurons. In the case before training, there is a lot of noise in the response (characterised by early firing of many neurons in the motor cortical layer). After training, the responses are less noisy, more spatially localised and in some cases have actually picked up some characteristics of the visual input pattern: see especially the response for pattern East after training which shows a stepped activation west to east.

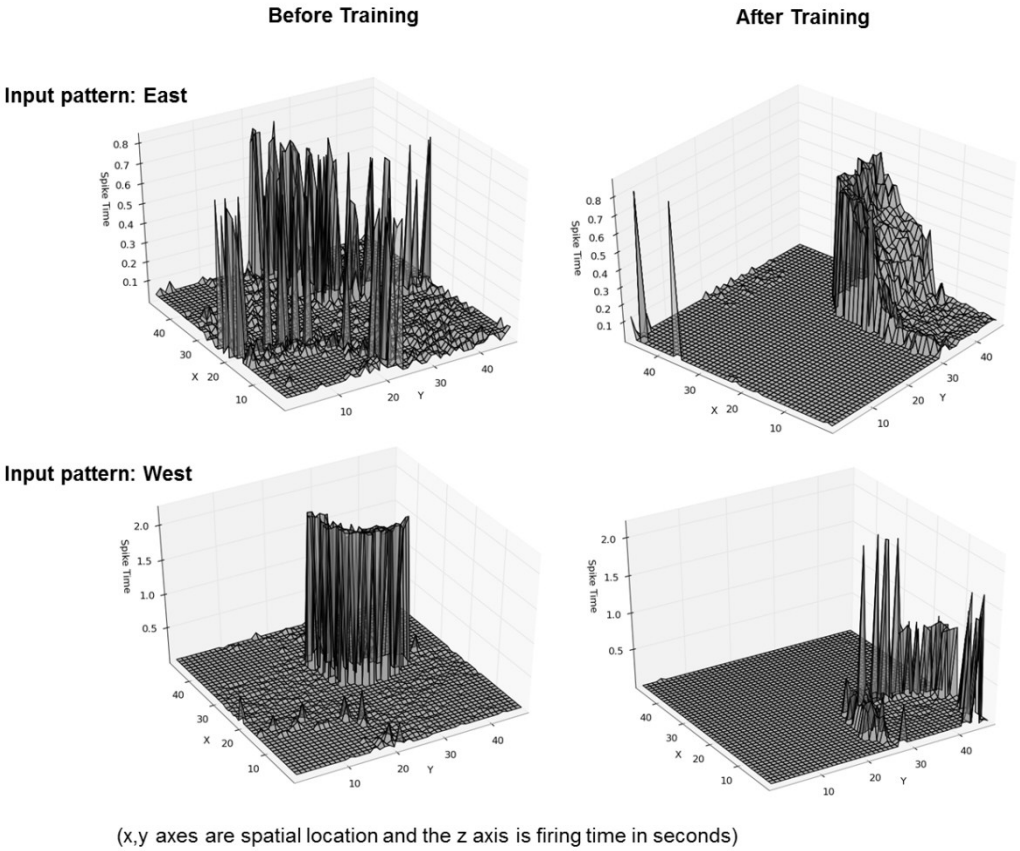


Figure 64 – Expt 12: Motor Map Response to Visual Input Before and After Training

Experiment 12 - Final Map Topography

Figure 65 shows the composite response of the motor map before and after training in spatial terms only. The results of training are qualitatively similar to those found for the motor map (compare Chapter 8, Figure 36). In Figure 65a (map before training) the responses to all the patterns overlap considerably. However, in Figure 65b (map after training) 68% of the cortical neurons have developed a preference for at least one of the 8 directions (there is some overlap in the response) and neurons with the same preference are mainly grouped in distinct patches. In the main, neighbouring patterns are also located near to each other, for instance patterns W (purple) and NW (white) are next to each other as are patterns S (orange) and SE (yellow).

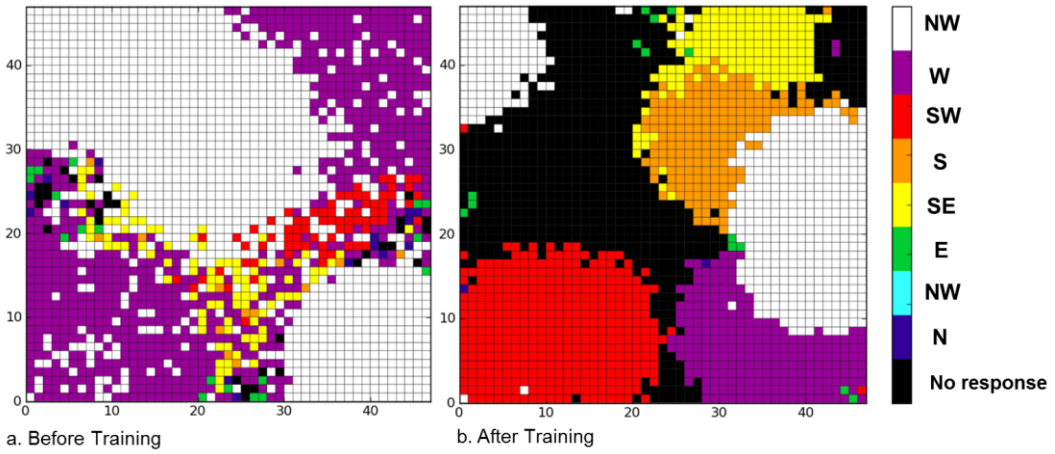


Figure 65– Expt 12: 48x48 Motor Map Topography

Experiment 12 - Rewiring Analysis

It was stated in Chapter 7, that rewiring was an essential part of the coupling process in order to ensure that, despite the initial random connectivity, the visual and motor cortical layers had access to the whole of each other's map and couple as and where the current activity required it. Rewiring is also important in ensuring that only useful connections are maintained and so as part of experiment 12 the total number of connections (excitatory and inhibitory) created and pruned was collected during the coupling training process. Figure 66 shows the total number of active connections per

neuron before and after training. Figure 66a shows the state of the connectivity in the initial network for the first 1000 visual cortical neurons. These have a count of between 850 and 1000. Figure 66b shows that after training, the maximum count is about the same but a large proportion of the neurons have counts a lot lower. Table 20 shows the rewiring statistics.

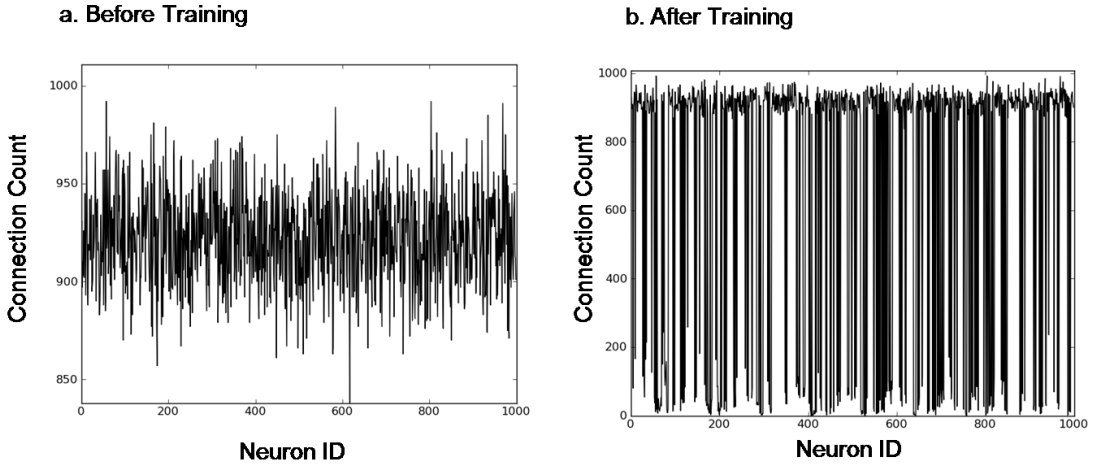


Figure 66 – Expt 12: Connection Counts Before and After Training

Starting connection count	12399499
Finishing connection count	10456077
Total pruned over 10 cycles	1943422
Total Created over 10 cycles	444117

Table 20 – Expt 12: Rewiring Statistics

In terms of overall connectivity, the total number of connections is reduced by about 16% although both creation and pruning occurred throughout training. The amount of pruning and creation relative to the overall number of connections is not as high as that found for the standalone motor map (see Table 16, Chapter 9, Section 9.6).

11 Discussion

11.1 Overview

This chapter presents the final discussion and conclusions of the research work. Firstly, Section 11.2 gives a summary and evaluation of the key results from Chapters 8-10, with emphasis on comparison to previous work especially where extensions and/or improvements have been made. Section 11.3 states how the research contributions stated in Chapter 1 have been met. Lastly, Section 11.4 summarises potential improvements to the current work and suggests directions for future work.

11.2 Evaluation of Results

Cortical Map Development

The motor system architecture was based upon the work of Marian (2002) which extended SOFM theory, previously almost exclusively used for visual map modelling, to the motor domain using spiking neural networks. In fact, Marian's work was quite generic as the inputs were manufactured and designated as directions around a compass, but could have represented any modality. In the current work, the first version of the motor system (16x16 cortical layer) was a replication of the work of Marian (2002) with several improvements with respect to making map training more autonomous. Firstly, the incorporation of a fixed, Gaussian neighbourhood function (originally due to Pham et al., 2006) to replace the normal SOM neighbourhood which requires a reduction schedule. Secondly, a common set of plasticity rules for both lateral excitatory and inhibitory connections based upon standard STDP with a weight dependent component to avoid the need for hard limiting or global normalisation. The benchmark experiment for this first version of the motor map (Chapter 8, Section 8.2) confirmed that a directionally selective, topographic map was produced after approximately 800 pattern presentations. As in a regular SOM, the afferent connections between input and output layers learned the characteristics of particular input pattern vectors, so the output

(cortical) neurons developed a preference for that pattern. In addition, analysis of the entire cortical layer response after training showed that the majority of neurons in the output layer responded preferentially to at least one of the input patterns, and the overall response to each pattern is distinct in both spatial (responding neuron) and temporal (neuron firing time) terms. These results were also confirmed in several other versions of the motor map: the scaled up version (48x48 cortical layer), the humanoid simulation version and the version with adaptive plasticity.

The visual system was based upon the previous works of Shon et al. (2004) and Wenisch et al. (2005) which had developed directionally selective visual maps using spiking neural networks. A crucial element present in both these works was the use of a particular form of asymmetric STDP rule which contributed to the development of directional selectivity. A novel addition in the current work is the use of the DVS 128 silicon retina camera as input which represents real moving images directly as spike events. In Shon et al. and Wenisch et al. (and in fact in the majority of previous works) the input was solid, rectangular moving bars either presented as a sequence of images or generated mathematically. The results of Wenisch et al. (2005) showed that directional selectivity in the lateral weights develops after training with repeated presentations of a single pattern and these results were reproduced to some extent in the current work in Experiment 4 (Chapter 8, Section 8.5). This experiment also showed that directional selectivity develops in the afferent weights. In addition, the analysis was extended by investigating the overall cortical response to presentations of the preferred and null patterns. This showed that the directional selectivity in the response manifests as a suppression of the activity from the null direction. A visual benchmark experiment (Chapter 8, Section 8.6) showed that, after training with the full pattern set, a topographic, directionally selective map representing 8 directions of motion was formed with the majority of neurons responding preferentially to at least one direction.

Surprisingly, it was found that with this setup a visual map reminiscent of the direction selective and orientation selective maps shown in Chapter 2, Figures 4 and 5 was present even before training. The training process appears to merely refine the response to represent the patterns actually seen and the overall cortical response is reduced resulting in a sparser response with less overlap. Plots of the spatiotemporal response for individual patterns showed that the reduction in activity allows the response to represent the characteristics of direction of movement more clearly. These results fit well with the previous observation that the directional selectivity arises by a reduction in response to the null direction as a result of the action of the asymmetric learning rule. The findings of Shon et al. (2004) also noted the role of inhibition in learning to allow representation of more than one direction of motion, however their model was confined to one dimension and only learned two directions of motion. To this author's knowledge no other works have noted the phenomenon of a pre-existing map before any training has occurred. However, it is known from experimental work that rudimentary visual maps are indeed present at eye-opening/birth to various degrees in different mammalian species. Their existence has been attributed to development via spontaneous activity (Rochefort et al., 2011; Espinosa and Stryker, 2012; Hunt et al., 2012). Several previous researchers have successfully modelled the creation of visual maps using only spontaneous activity, for example, Wenisch et al. (2005) and Miikkulainen et al. (2005). It should be noted however, that in real cortical visual maps the spontaneous activity is working in conjunction with molecular signalling to establish connectivity between the LGN and V1 (Espinosa and Stryker, 2012) which could imply that it is the connectivity that is important for the existence of the initial map, and the process of making this connectivity is facilitated by spontaneous activity. The discussion section of Hunt et al. (2012) cites the work of Kaschube et al. (2010), which looked at orientation selectivity from an evolutionary perspective across several mammalian species and found that

intrinsic network self-organisation may be important in the development of similar orientation maps across different species. Hunt et al. also propose that the self-organisation may somehow predefine the receptive field organisation which is then refined by both intrinsic spontaneous and external activity. In the current work it is not clear whether it is solely the initial connectivity (in terms of the receptive field size or organisation from the Retinal to Output layers and the lateral recurrent connection profiles) or the nature of the input (spatio-temporal pattern of individual spikes) that is responsible for the pre-existing map. Comparing the methods used here to previous works the major difference is indeed the nature of the visual input and a possible explanation for the map is that the input spike events coupled with the lateral recurrent activity result in small, sequential bursts of activity which break the symmetry of the randomly initialised network to cause a directionally selective response.

Adaptive Plasticity

The Plasticity Resource (PR) was introduced in Chapter 5 as an alternative to a tradtional SOM learning rate reduction. Its purpose was to monitor and control map training autonomously, guided only by the input activity. Previous researchers have already noted the inconvenience of defining reduction schedules for the traditional SOM learning rate and proposed various schemes for an adaptive learning rate rather than a predefined schedule (Shah-Hosseini and Safabakhsh, 2000; 2001; Berglund and Sitte, 2006; Miyoshi, 2005; 2007; 2008; Berglund, 2010; Shah-Hosseini, 2011). However, none of these previous works used spiking neural networks or considered the possibility that the training input could be other than a vector of values which directly modify afferent connection weight vectors. In the current work the motor input is a vector of spike times but the visual input is a stream of spike events from the DVS camera. A bio-inspired approach suitable for both types of input has been used and views the input activity in terms of the LTP / LTD occurring globally over the afferent connections. It

has been inspired by the notion of consumption of Neurotrophic factor (NTF) in the development of biological cortical maps. Experiments were done for both the 16x16 and 48x48 motor map and also the visual map (Chapter 9, Sections 9.2-9.5) which showed firstly that the basic characteristics of the benchmark results were not affected by the addition of the PR to the learning rules. In addition an experiment using a new dataset and online random pattern presentation, where the number and composition of input patterns required to train the map was unknown demonstrated that the PR allows monitoring of the training and indicates when the training should be stopped. The second aspect of Adaptive Plasticity used in the current work was synaptic rewiring. The rewiring scheme described in Chapter 5, Section 5.3 was incorporated into the motor map and visual map training to work alongside the Plasticity Resource. It was found that the map results were qualitatively similar to those produced without rewiring but the connectivity in the final networks is sparser. but the connectivity in the final networks is sparser. This is important as it shows that rewiring can help the network adapt to the current activity better and only preserve connections which really matter in representing the input, thus reducing the number of connections which need to be maintained: a very important benefit when implementing such systems in neuromorphic hardware. Despite the overall reduction in connectivity the results collected during the training process showed a high level of both pruning and creation ongoing throughout training. Recent reviews of adult structural plasticity (Butz et al., 2009; Gilbert and Li, 2012) confirm that there is experimental evidence for constant forming or breaking of synapses in the adult brain and that the forming and breaking of synapses is directly influenced by the action of LTP and LTD. Moreover, synapses can be in semi-stable or unstable states which form and break on relatively short timescales. The issue of the rate of rewiring as compared to weight changes was raised in Chapter 9. In the current work using only the temporal correlation of firing, STDP controlled weight changes and a

hard pruning threshold results in functional and structural plasticity operating at comparable rates. However, the relatively fast fluctuations in creation and pruning may not reflect the rate of *permanent* creation and pruning (i.e. connections which are pruned and stay pruned or created and persist): more modelling work would need to be done to establish what this rate is and whether it is slower than the rate of weight changes as suggested in the review of Chklovskii et al., 2004.

Simulation

Although it has not been possible to fully realise the cortical map training on a real robot some useful work has been done in simulation which should help to inform future implementation. In particular the issue of 'sensorimotor integration' or how the neural processing can be integrated with a motor system to generate behaviour. Two quite different modelling studies were done in simulation. The first a Computational Neuroethological model of arachnid prey localisation which extended an existing neural model to make turning and walking motor behaviours based upon the predictions of the neural processing. This relatively simple prototype system relied upon a hardwired neural architecture but set the scene for the next step which was the integration of the cortical motor map training with a humanoid robot simulation. Motor map training (with and without Adaptive Plasticity) using the humanoid simulation as the environment gave qualitatively similar results to the original experiments, although run times were longer due to the simulation overhead. More importantly, the humanoid simulation also provided a testing environment and helped to answer the question of how the motor map response could be used to generate an orientation behaviour for the robot.

Analysis of map responses as spike trains

Chapter 2, Section 2.5 discussed some previous works which had investigated sensorimotor coordination / control using SOMs. Of the two which had actually used

spiking neural networks (Marian, 2002; Alamdari, 2005) neither had really considered how the trained map response could be used to directly generate a behaviour. In the case of Marian, the analysis of the resulting maps was only taken as far as necessary to establish that motor and visual topographic maps had been created and that they could be coupled together to coordinate the visual and motor activity. In Alamdari, the SOM learned to represent obstacle locations, but a traditional path planning step was required to make use of this information. In the current work, plots of the spatiotemporal responses of the trained motor and visual maps appeared to indicate that the maps had learned a distinct response for each input pattern. A method incorporating the van Rossum metric (van Rossum, 2001) was used to compare the spatial (neuron ID) and temporal (neuron firing time) aspects of the responses in a quantitative way. This novel application of the van Rossum metric treated a map response as a collection of spike trains over all the cortical neurons and allowed comparison to the responses of the exemplar patterns. The method worked efficiently for a range of response sizes (up to as many as 2000 responders in the case of the visual map). Experiments with the 16x16 motor map showed that the method could distinguish between the cortical responses of the trained map to the 8 different input patterns. In the case where deliberately ambiguous input was presented (i.e. midway between two directions), as the measure is distance based it would naturally select the closest pattern. The most successful application of the response classification was demonstrated with motor map testing in the humanoid simulation as it showed how the map response could be used to generate a behaviour. Experiments using the simulation with the van Rossum analysis demonstrated that the motor map response could be decoded to predict what the sensory input was and execute the correct motor movements. Specific experiments using ambiguous input showed that the system was able to make a reasonable decision resulting in an orientation movement that corresponded to the closest exemplar direction.

For the visual system the van Rossum analysis method was not tested as thoroughly as only one set of patterns were used without any perturbation. Nonetheless the method was shown to work correctly as it could distinguish unambiguously which input pattern had generated a particular map response.

Coupling

An important advance delivered by the work of Marian (2002) was to suggest a method for learning coupling of cortical maps using STDP and the motor babbling learning sequence of Kuperstein (1998). This was quite a different approach to other visual-motor coordination works before this time which usually involved directly learning the kinematics or dynamics of the motor system (for example Ritter et al., 1989; Metta et al., 1999). The current work has used the same approach as Marian and extended it by using larger sized maps (48x48 motor and 116x116 visual cortices) and more realistic visual spiking input provided by the DVS 128 silicon retina. Another important extension has been the use of synaptic rewiring in the coupling learning. In Marian's work the maps were small enough that full visual to motor connectivity was possible (16x16 motor and 18x18 visual cortices), however in the current work with larger maps this was not feasible. In the coupling process it is necessary that the visual cortex has access to all areas of the motor directional map and so, in keeping with the thread of autonomous activity dependent development and learning in this work, the solution was to allow the coupling process to create and prune connections as required dependent upon the co-activation of the two cortical maps. The experimental results (Chapter 10) showed that in the initial coupled state before training, the motor output when stimulated solely by visual input is not particularly directionally selective and there is a lot of noise in the response. After training a directionally selective topographic map has developed and in some cases, the motor response has picked up the temporal characteristics of the moving visual input. Comparing the motor topographic map produced via coupling to

the original (48x48) map one can see that the maps are quite different. In the coupled version there is more overlap and some patterns are not distinguishable in purely spatial terms. However the temporal characteristics are much more important here than in the original motor map as the visual input has very specific temporal characteristics which influence the motor response. Comparing the coupling results to those of Marian (2002) there are some major differences. Her results indicated that the visual input learned to directly reproduce the original motor response, in the sense that the responding population of neurons is similar. However, in the discussion section of this work it is mentioned that in the coupled case the neurons response tuning is much broader (neurons respond to more directions than in the pure motor case) and also that neurons become responsive in the visual input case which were previously silent in the motor only case. It is difficult to make a proper comparison to the current work as Marian's analysis only shows the response for one pattern and does not include a topographic map. Differences in the results are most likely due to changes made in the current work, which can be summarised as follows:

- In Marian (2002) the visual input was manufactured Gaussian bars presented in a very controlled way in the coupling learning procedure. In the current work the input is sequences of real spike data from the DVS camera which vary in the number of neurons they activate and the duration of the movement
- In Marian the visual map was pre-created so would have been an 'ideal' map, whereas in the current work the visual map was developed using the random presentation of inputs
- The maps in the current work are considerably larger
- The current work included rewiring

In the current work the results can be interpreted as the visual modality modulating the motor rather than directly reproducing it. In this point of view the initial uncoupled

motor map can be seen as still relatively coarse (for example, in an infant would represent inaccurate and ballistic movements). The visually coupled motor map is different because the motor response is actually changed by the visual input producing a coordinated response (refer back to Chapter 1, Figure 1.1 for the developmental stages envisaged in this work). It should be noted that in the motor map the original motor response is in some sense still present and can be reproduced by stimulation from the motor modality only. The coupling to the visual modality is also only one possible modulation: there is the possibility of different responses due to modulation from other modalities or even concurrent input from more than one modality, for instance visual and auditory.

As well as allowing coupling of much larger maps, the benefit of including rewiring in the coupling learning is that it results in a lower final connectivity between the cortical maps, which is in agreement with the results of allowing rewiring within individual maps. It is likely to also be of benefit in allowing 'recovery from lesion' studies with coupled maps, although that avenue of work has not been pursued here.

11.3 Summary of Research Contributions

The first contribution has been the creation of a biologically-inspired developmental framework encompassing both motor and visual cortical map creation as well as allowing coupling between the two types of map to achieve a basic visuomotor coordination.

Section 11.2 summarised the methods that have been used to achieve this which are based upon previous work with several extensions and novel additions. The results have shown that in both the motor and visual cases the training process produces directionally selective maps which exhibit distinct responses to the different input patterns.

The second contribution has been the development of a methodology that enables

internal regulation of cortical map development and dispenses with predefining traditional aspects of SOM training such as learning schedules, neighbourhood reduction schedules and the amount of training data. This is particularly important for the future application of these methods to learning in autonomous robots. Section 11.2 summarised this methodology, which is based upon features of real biological development and improves upon previous works by using spiking neural networks and an activity-dependent process based directly upon the action of LTP and LTD. The method is demonstrated to work for both motor and visual maps which have quite different input data formats.

The last contribution is the use of a bio-inspired neuromorphic device in visual cortical feature map training. A methodology has been developed to use the DVS 128 Silicon Retina camera as input which provides real-world input directly as spike events. Section 11.2 described how the experiments confirmed that the map development system with this input could produce basic directional selectivity mediated by asymmetric afferent and lateral weight changes and also that, with no extra mechanisms, a full topographic directional map was produced.

11.4 Future Work

From the previous discussion, there are several aspects of the current work that could be improved. With respect to the visual system, one major deficiency is the limited number of training sequences that were used: only one for each exemplar direction. Ideally, several different sequences for the same direction of movement should have been captured from the DVS camera and when a direction was selected, one of these instances should have been randomly presented. Having a bigger range of visual inputs and also test sequences which represented directions in between the exemplar directions would also strengthen the van Rossum metric analysis. The work done with the humanoid simulation did not incorporate any visual input although this could have been

technically possible, either using logged DVS sequences or even live camera input (see Appendix B which discusses additional work feeding live camera into neural system). However, it was felt that it was perhaps more useful to defer this until at least the visual system was implemented in neuromorphic hardware.

In the coupling work, the results did not exactly reproduce those of Marian (2002) and several possible reasons have been proposed in Section 11.2. It is likely that the introduction of rewiring may be significant here but it has not been possible to test the coupling without rewiring due to the size of maps used. This could possibly be remedied by repeating the work using similar sized maps to Marian in scenarios with and without rewiring.

In terms of future extensions to the work, the most important is undoubtedly some form of hardware implementation. Although the ultimate aim is to implement these bio-inspired methods on board a robot, it would not be particularly useful to just use the work on a PC communicating with a robot. Instead, a neuromorphic implementation using the SpiNNaker technology is planned and this research has been done with this in mind at the outset. The software development has been done in Python, mainly in the Brian spiking neural simulator and as such is already fit to transfer to a PyNN implementation and then a PyNN-SpiNNaker implementation. The basic neural models and network architectures used present no issues with respect to SpiNNaker implementation. The use of the DVS camera as input to the visual system does present some problems, as the rate of data generated is too much for the SpiNNaker system to cope with. Therefore modifications to the current work would require downsampling or preprocessing of the DVS spikes before they are applied to the network. This is a known issue and one successful solution has been to use an FPGA board as an interface to process and downsample the spikes (Galluppi et al., 2012).

As mentioned elsewhere in this thesis, rewiring plasticity is still a relatively unexplored

area in the domain of Neurorobotics. The current work has shown some benefits of using rewiring, in the case of ensuring that connectivity is a sparse as possible, but there is substantial room for taking this further, particularly in robustly showing that allowing it makes a map better able to respond to changing input than without it. Rewiring plasticity could be an interesting avenue to explore in neuromorphic implementations. It has obvious benefits with respect to achieving optimum sparseness so that only a minimal number of connections need to be maintained. There could also be a possibility of using the 'recovery from lesion' properties to improve fault tolerance. For instance if a processor or chip malfunctions the network would in theory be able to prune defunct connections and create new ones to still active cores/chips. Some recent work has also shown that rewiring may play a part in homeostatic processes (Butz et al., 2009) which could be another of avenue for exploration in neuromorphic modelling.

APPENDICES

A Creation of the Motor Input Patterns

A.1 Overview

Chapter 3, Section 3.4 introduced the form of the input patterns used for the motor map training. This appendix gives more details of how the 8 directional patterns were constructed and how the pattern set was validated before training. The process of training dataset creation is also described. Chapter 9, Section 9.2.3 described an experiment that was done to validate the Adaptive Plasticity methods by training with a 'new' pattern set which consisted of robot poses from the humanoid simulation, and the design, creation and validation of these data are also documented here.

A.2 Design of the directional patterns

The form of the input patterns was guided by those described in Marian (2002). However, as a full description of how the patterns were constructed is not given in that work, the method of construction for the current work is inferred based upon the description given, and later validated (see section A.4).

The directional pattern set consists of 8 exemplars (N, NE, E, SE, S, SW, W, NW) and each pattern consists of two vectors consisting of 16 elements. The first vector holds neuron IDs and the second neuron firing times. Of the 16 elements, 4 are 'salient' (i.e. they provide actual information) and the rest are 'noise'. Salient neurons have firing times close to the base (integration) time which is 9 milliseconds, whilst noise neurons fire at much earlier times. Patterns that are 'adjacent' (for example, directions N and NW) share 2 out of 4 salient bits of information (i.e. the same neuron firing at the same time) whilst patterns that are opposite (for example, directions N and S) do not share any salient information. In the current work the following process was used to create the 8 exemplar patterns:

1. Generate 8 sets of 4 ‘salient’ firing times in the range 7-9 ms (i.e. they fire close together near t = 9ms).
2. Generate 8 sets of 12 ‘noise’ firing times in the range 0-3 ms (i.e. they fire closer to t=0ms with times that do not overlap with salient values)
3. Assign neuron IDs (in the range 0-15) to the firing times and ensure that adjacent directions have two elements in common and that opposite directions do not have any information in common.

Note that in the generation of the firing times in steps 1 and 2 the precision of the values is 0.1ms to match the timestep used in the Brian simulations.

A.3 Creating training datasets

For training with the initial version of the motor map predefined datasets were required. These were created by making 20 instances of each of the 8 exemplar directional patterns perturbed from the original values. Perturbing of the exemplar patterns is performed as follows:

1. Only spike times are perturbed
2. Noise values are adjusted by a random value within the range (-1.0,1.0). Minus values are clipped to 0ms
3. Salient values are adjusted by a random value within the range (-0.5,0.5). Values greater than 9.0 are clipped to 8.9ms

A total of 160 patterns were written in a random order to the training dataset.

A.4 Validation of the directional patterns

It was important to establish that the exemplar patterns did actually reflect the characteristics of motor directions that they were supposed to represent, especially that the patterns were similar/dissimilar in the correct way and that they were separable so

the SOM training process would be able to represent them as a 2D map. This was done in two ways. Firstly, the Euclidean distance was calculated, using the spike times for each neuron, between each pair of patterns. This was then normalised so the distances were in the range 0-1. This data is given in Table 21.

	N	NE	E	SE	S	SW	W	NW
N	0.00	0.42	1.00	0.92	0.91	0.92	0.61	0.30
NE		0.00	0.45	0.92	0.86	0.88	0.57	0.63
E			0.00	0.40	0.84	0.93	0.87	0.85
SE				0.00	0.32	0.98	0.62	0.59
S					0.00	0.37	0.64	0.66
SW						0.00	0.43	0.65
W							0.00	0.00
NW								0.00

Table 21 – Normalised Distances Between Exemplar Motor Patterns

Table 22 summarises the distances, and average distance between patterns $^{1}/_{8}$ **,** ¼, ¾ and ½ a compass turn apart. This indicates that, as required, patterns that are closer to each other in terms of compass directions are more similar (lower scores) and patterns that are opposite are less similar (higher scores).

The second validation was a Fisher's Linear Discriminant Analysis (FDA) performed on one training dataset (i.e 160 patterns perturbed from exemplars as described in Section A.3) using the SPSS software. Figure 67 shows the final graphical clustering of the data which indicates (with perhaps the exception of exemplars 7 and 8) that the classes are well separated.

1/8 turn apart		¼ turn apart		¾ turn apart		Opposite	
N-NE	0.42	N-E	1.00	N-SE	0.92	N-S	0.91
NE-E	0.45	E-S	0.84	SE-W	0.62	E-W	0.87
E-SE	0.40	S-W	0.64	W-NE	0.57		
SE-S	0.32	W-N	0.61	NE-S	0.86		
S-SW	0.37	NE-SE	0.92	S-NW	0.66		
SW-W	0.43	SE-SW	0.98	NW-E	0.85		
W-NW	0.00	SW-NW	0.65	E-SW	0.93		
NW-N	0.30	NW-NE	0.63	SW-N	0.92		
Average	**0.34**		**0.78**		**0.79**		**0.89**

Table 22 – Summary of Distances Between Patterns for Fixed Compass Turns

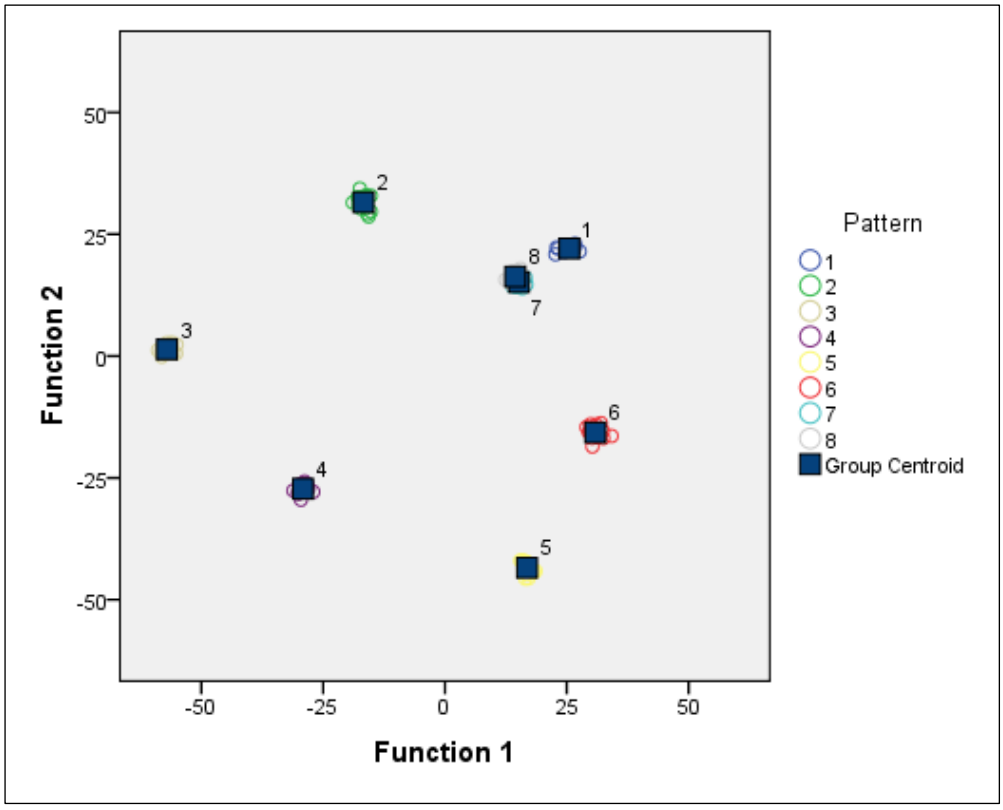


Figure 67 – Clustering of the Motor Directional Patterns using FDA

A.5 Pose patterns from the humanoid simulation

For one of the Adaptive Plasticity experiments (see Chapter 9, Section 9.2) a new pattern set was created using 4 robot poses from the humanoid simulation (Left Leg Kick, Left Arm Point, Right Leg Kick, Right Arm Point). In this case the starting point was 16 servomotor angles which represented the changes that needed to be made for the simulated robot to make the relevant pose from a base standing position. These values naturally contained salient and noise data as the poses involve a different subset of the servomotors. In order to convert these angles into spike times the Linear Temporal Encoding of Maass (1997) was used with the assumption that large movement → large angle change → smaller latency. The conversion is done using equation (35).

$$t = T_{int} * (x/MaxVal) \quad (35)$$

Where:

T_{int} is a reference time marking the end of arrival of all of the inputs.

x is the input angle

MaxVal is the largest angle to be encoded

t is the resulting spike time

This equation ensures that all of the input angles are mapped to the range 0-T_{int}.

One difficulty was that, as with the motor patterns, the spike times needed to have a precision of 0.1ms and it is possible that the method generates the same spike time for different angles at this precision. To deal with this, non-unique spike times are accepted if the original data values are, in fact the same (to 2 d.p.) , but where they are not, extra work is needed to allocate unique spike times based upon the rank order of original data values. The method used is as follows:

1. Sort the input data values in ascending order

2. Assign a ‘rank’ with identical data values taking the same rank.
3. Iterate through the list of values calculating t to 1 d.p. using Equation (32)
4. Check the generated spike time against that of the previous item. Where the time and the rank are the same as the previous item, leave the spike time as is, otherwise ensure spike time is different from previous item (increment by 0.1).

A.6 Validation of the pose patterns

The same analysis as described in Section A.4 for the directional patterns was done.

Table 23 shows the normalised distances between the exemplar pose patterns.

	Left Kick	Left Point	Right Kick	Right Point
Left Kick	0.00	0.58	0.08	0.77
Left Point		0.00	1.00	0.00
Right Kick			0.00	0.89
Right Point				0.00

Table 23 – Normalised Distances Between Exemplar Pose Patterns

These data show that kick and point patterns are viewed as being quite dissimilar, whereas left and right point and left and right kick are similar. This situation is not quite as good as for the directional patterns but is to be expected as the directional patterns were specially manufactured to be similar/dissimilar in specific ways, whereas the pose data has been taken as is from the simulation. The Fisher’s Linear Discriminant Analysis (FDA) was performed on a set of 100 patterns (25 of each of the 4 exemplars randomly perturbed as described in Section A.3) using the SPSS software.

Figure 68 shows the final graphical clustering of the data which indicates that the pattern classes are well separated.

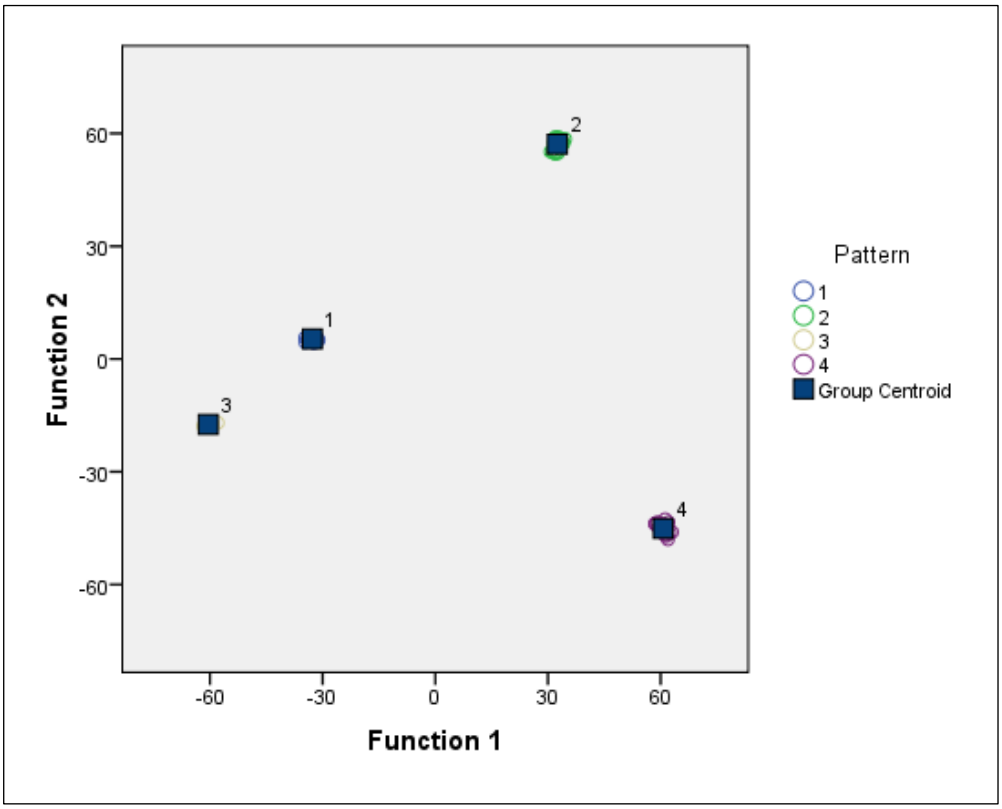


Figure 68 – Clustering of the Motor Pose Patterns using FDA

A.7 'Online' training

In the experiments described in Chapter 9, Sections 9.2, 9.3 and 9.4 predefined training datasets were not used. Instead patterns were randomly selected and presented as the point here was that the Adaptive Plasticity methods would control the training and indicate when it had completed. In this case, the method used was to generate a random integer in the range [1-8], select the relevant exemplar pattern and perturb the spike times using the same method described in Section A.3.

B Dynamic Training with the DVS 128 Silicon Retina

B.1 Overview

Chapter 4, Section 4.4 explained that whilst the main experiments with visual map training were done using pre-recorded sequences from the DVS camera, a mechanism had been created to take input directly from the live camera and this is described in this Appendix. Section B.2 describes the method of obtaining and processing the raw camera events using the jAER Java API. Section B.3 describes the system architecture that was created to handle the continuous input of camera events and is based upon the Visual Map training process described in Chapter 4, Section 4.6.

B.2 Processing Raw Camera Events

The handler for the camera has been written as a Java program which uses the following classes from the jAER API (jAER SourgeForge wiki, 2012):

```
net.sf.jaer.aemonitor.AEPacketRaw
net.sf.jaer.eventio.AEUnicastOutput
net.sf.jaer.hardwareinterface.HardwareInterfaceFactoryInterface
net.sf.jaer.hardwareinterface.usb.cypressfx2.CypressFX2DVS128Har
dwareInterface
```

It takes raw camera events and outputs them as UDP packets by the following process:

1. Get an interface to the first camera device found

```
HardwareInterfaceFactoryInterface instance =
USBIOHardwareInterfaceFactory.instance();
CypressFX2DVS128HardwareInterface dvs128interface = in-
stance.getFirstAvailableInterface();
```

2. Get event packets as type AEPacketRaw

```
AEPacketRaw events =
dvs128interface.acquireAvailableEventsFromDriver();
```

3. Open a UDP Socket

```
unicastOutput = new AEUnicastOutput();
unicastOutput.setPort(portnum);
unicastOutput.setSequenceNumberEnabled(false);
unicastOutput.setBufferSize(buffsize);
```

```
unicastOutput.open();
```

4. Write the events to the UDP socket

```
unicastOutput.writePacket(events);
```

B.3 The Dynamic Training Process

Figure 69 gives an overview of the process of supplying events from the live DVS camera to the training process. Once the Java camera interface is running, events are sent as UDP packets to a designated port. The Brian neural code was extended to incorporate a UDP server class which is started first and runs on a separate thread from the neural processing. The UDP server has two functions: 1) to continually poll for camera events coming in on the designated port and decode them into lists of neuron IDs and spike times, 2) to send these lists of events to the Neural class which executes on a separate thread.

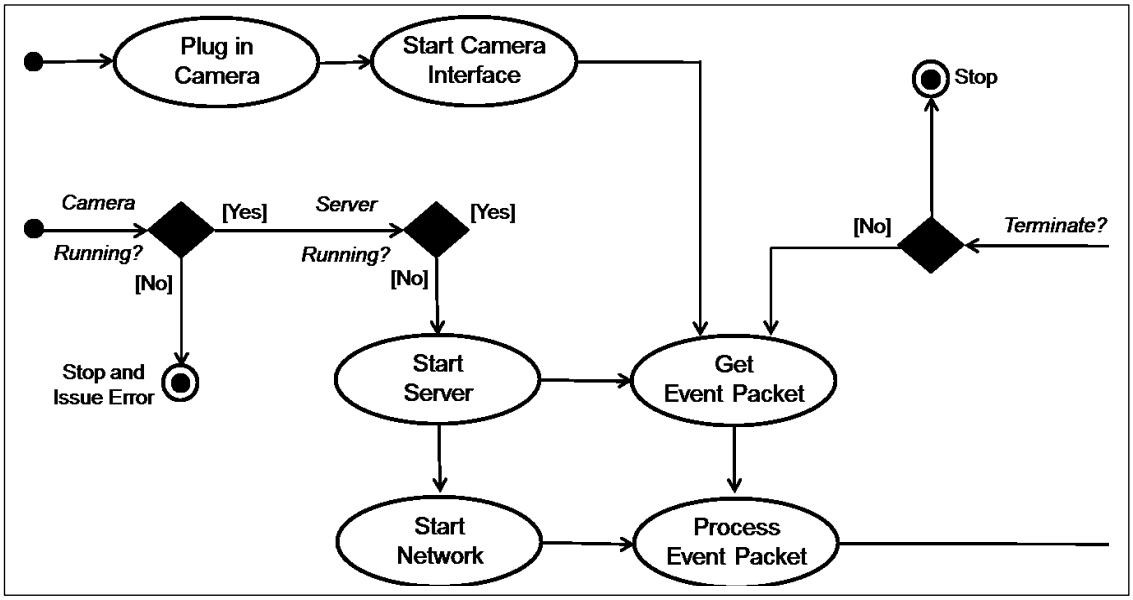


Figure 69 – An Overview of the Training System Using a Live DVS Camera

Decoding the UDP packets uses the same methods as described in Chapter 4, Section 4.4 (code from the Brian module *experimental/neuromorphic/AER.py)* and as for the logged data, OFF events are filtered out. Modifications were made to the original neural processing code to accept a list of spike times when it is called from the UDP server.

The neural code calculates the required run time from this list and from then on operates in the same way as if the input had been a logged aedat file. Once neural processing has completed its thread is terminated. The UDP server monitors whether the neural thread is running or not and waits for it to complete before reading more camera events.

References

Abbott, L.F. (1999) 'Lapicque's introduction of the integrate-and-fire model neuron (1907)', Brain Research Bulletin, **50** (5-6),: 303–304,

Adams, S.V., Culverhouse, P.F, Wennekers, T., Bugmann, G. and Denham, S. (2010) 'A Sensori-Motor Model of Arachnid Prey Localisation' in: *Proceedings of the 11th Towards Autonomous Robotic Systems Conference (TAROS 2010)*, Plymouth, UK: 1-6.

Adams, S.V., Wennekers, T., Bugmann, G., Denham, S. and Culverhouse, P.F (2011) 'Application of Arachnid Prey Localisation Theory for a Robot Sensorimotor Controller', *Neurocomputing*, **74**(17): 3335-3342

Alamdari, A. (2005) 'Unknown Environment Representation for Mobile Robot Using Spiking Neural Networks' in: *Proceedings of the WEC, Transactions on Engineering, Computing and Technology*.

Alnajjar, F. and Murase, K. (2008) 'A Simple Aplysia-Like Spiking Neural Network to Generate Adaptive Behavior in Autonomous Robot', *Adaptive Behavior*, **16**(5): 306-324.

Arbib, M.A. and Liaw, J. (1995) 'Sensorimotor transformations in the worlds of frogs and robots', *Artificial Intelligence,* **72**: 53-79.

Arbib, M., Metta, G. and van der Smagt, P. (2008) 'Chapter 64: Neurorobotics: from vision to action' in: B.Siciliano and O.Khatib (eds.) *Springer Handbook of Robotics*: 1453-1480.

Archambault, P. S., Ferrari-Toniolo, S. and Battaglia-Mayer, A. (2011) 'Online Control of Hand Trajectory and Evolution of Motor Intention in the Parietofrontal System', *The Journal of Neuroscience,* **31**(2): 742-752.

Bamford, S. (2009) 'Synaptic Rewiring in Neuromorphic VLSI for Topographic Map Formation', PhD thesis, University of Edinburgh.

Bednar, J. and Miikkulainen, R. (2000) 'Tilt aftereffects in a self-organizing model of the primary visual cortex', *Neural Computation,* **12**(7): 1721-1740.

Bednar, J. and Miikkulainen, R. (2003) 'Self-organization of spatiotemporal receptive fields and laterally connected direction and orientation maps', *Neurocomputing*, **52-54**: 473-480.

Berglund, E. and Sitte, J. (2006) 'The Parameterless Self-Organizing Map Algorithm', *IEEE Transactions on Neural Networks* **17**(2): 305-316.

Berglund, E. (2010) 'Improved PLSOM Algorithm', *Applied Intelligence* **32**(1): 122-130.

Bi, G. and Poo, M. (1998) 'Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type', *The Journal of Neuroscience***, 18**(24): 10464-10472.

Bizzi, E., Giszter, S., Loeb, E., Mussa-Ivaldi, F. A. and Saltiel, P. (1995) 'Modular organization of motor behavior in the frog's spinal cord', *Trends in Neurosciences*, **18**(10): 442-446.

Bohte, S., La Poutre, H. and Kok, J. (2002) 'Unsupervised clustering with spiking neurons by sparse temporal coding and multilayer RBF networks', *IEEE Transactions on Neural Networks,* **13**(2): 426-435.

Bosking, W.H., Zhang, Y., Schofield, B. and Fitzpatrick, D. (1997) 'Orientation Selectivity and the Arrangement of Horizontal Connections in Tree Shrew Striate Cortex', *The Journal of Neuroscience*, **17**(6): 2112-2127.

Boulenger, V., Hauk, O. and Pulvermüller, F. (2009) 'Grasping Ideas with the Motor System: Semantic Somatotopy in Idiom Comprehension', *Cerebral Cortex,* **19**: 1905-1914.

Brette, R., Rudolph, M., Carnevale, T., Hines, M., Beeman, D., Bower, J.M., Diesmann, M., Morrison, A., Goodman, P.H., Harris, F.C., Zirpe, M., Natschläger, T., Pecevski, D., Ermentrout, B., Djurfeldt, M., Lansner, A., Rochel, O., Vieville, T., Muller, E., Davison, A.P., El Boustani, S. and Destexhe, A. (2007) 'Simulation of networks of spiking neurons: A review of tools and strategies'. *Journal of Computational Neuroscience,* **23**(3): 349-398

Brohan, K., Gurney, K. and Dudek, P. (2010) 'Using Reinforcement Learning to Guide the Development of Self-organised Feature Maps for Visual Orienting' in: *The Proceedings of the International Conference on Artificial Neural Networks (ICANN 2010)*

Brooks, R. (1990) 'Elephants don't play chess', *Robotics and Autonomous Systems*, **6**: 3-15.

Brownell, P. H. (1977) 'Compressional and Surface Waves in Sand: Used by Desert Scorpions to Locate Prey', *Science*, **197**: 497–482.

Brownell, P. H. and Farley, R. (1979) 'Orientation to Vibrations in Sand by the Nocturnal Scorpion Paruroctonus mesaensis: Mechanism of Target Localization', *Journal of Comparative Physiology A: Neuroethology, Sensory, Neural, and Behavioral Physiology,* **131**(1): 31-38.

Brownell, P. H. and van Hemmen, J. L. (2001) 'Vibration Sensitivity and a Computational Theory for Prey-Localizing Behavior in Sand Scorpions', *American Zoologist,* **41**(5): 1229-1240.

Burnod, Y., Baraduc, P., Battaglia-Mayer, A., Guigon, E., Koechlin, E., Ferraina, S., Lacquaniti, F. and Caminiti, R. (1999) 'Parieto-frontal coding of reaching: an integrated framework', *Experimental Brain Research*, **129**: 325–346.

Bushnell, E. (1981) 'The ontogeny of intermodal relations: vision and touch in infancy' in: R. Walk and H. Pick (eds) *Intersensory perception and sensory integration*, New York: Plenum Press: 5-37

Butz, M., Wörgötter, F. and van Ooyen, A. (2009) 'Activity-dependent structural plasticity', *Brian Research Reviews*, **60:** 287-305.

Carenzi, F., Gorce, P., Burnod, Y and Maier, M.A. (2005) 'Using generic neural networks in the control and prediction of grasp postures' in: *The Proceedings of the 13th European Symposium on Artificial Neural Networks (ESANN 2005)*: 61-66

Carpenter, G., Grossberg, S., Markuzon, N., Reynolds, J., and Rosen, D. (1992) 'Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps', *IEEE Transactions on Neural Networks*, **5**: 698–713.

Chan, V., Liu, S.-C. and van Schaik, A. (2007) 'AER EAR: A matched silicon cochlea pair with address event representation interface', *IEEE Transactions on Circuits and Systems I: Special Issue on Smart Sensors,* **54**: 48-59.

Changeux, J. and Danchin, A. (1976) 'Selective stabilisation of developing synapses as a mechanism for the development of neuronal networks', *Nature,* **264**: 705-712.

Cheung, C.L. (2008), clrobotsim application available from Google code site http://code.google.com/p/clrobotsim/

Chklovskii, D., Mel, B. and Svoboda, K. (2004) 'Cortical rewiring and information storage', *Nature,* **431:** 782-788.

Choe, Y. and Miikkulainen, R. (1998) 'Self-organization and segmentation in a laterally connected orientation map of spiking neurons.', *Neurocomputing,* **21**: 51-60.

Cliff, D. (1991) 'Computational neuroethology: a provisional manifesto' in: Proceedings of the First International Conference on Simulation of Adaptive Behavior, From Animals to Animats, MIT Press: 29-39.

Crespi, A., Badertscher, A., Guignard, A. and Ijspeert, A.J. (2005) 'Swimming and crawling with an amphibious snake robot' in: *Proceedings of the IEEE International Conference on Robotics and Automation*: 3035–3039.

Damper, R. I. and French, R.L.B. (2003) 'Evolving spiking neuron controllers for phototaxis and phonotaxis' in: Applications of Evolutionary Computation, EvoWorkshops.

Davies, S., Patterson, C., Galluppi, F., Rast, A., Lester, D. and Furber, S. (2010) 'Interfacing Real-Time Spiking I/O with the SpiNNaker neuromimetic architecture', in: *The Proceedings of the 17th International Conference on Neural Information processing (ICONIP 2010)*: 7-11.

Dayan, P. (2004), 'Pattern formation and cortical maps', *Journal of Physiology Paris,* **97**(4-6): 475-489.

Dayan, P. and Abbott, L.F (2001) 'Chapter 8, Plasticity and Learning' in: *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*, MIT Press: 281-329.

Dayan, P. and Abbott, L.F. (2001) Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems, MIT Press, Cambridge Massachusetts

Delbrück, T. (2008) 'Frame-free dynamic digital vision' in: The Proceedings of the International Symposium on Secure-Life Electronics, Advanced Electronics for Quality Life and Society: 21-26.

Edelman, G. (2007) 'Learning in and from Brain-Based Devices', Science, **318**: 1103-1105.

Elliott, T. and Kramer, J. (2002) 'Coupling an aVLSI neuromorphic vision chip to a neurotrophic model of synaptic plasticity', *Neural Computation*, **14**: 2353-2370.

Elliott, T. and Shadbolt, N. (1998) 'Competition for Neurotrophic Factors: Mathematical Analysis', *Neural Computation,* **10**(8): 1939-1981.

Elliott, T. and Shadbolt, N. (1998) 'Competition for Neurotrophic Factors: Ocular Dominance Columns', *The Journal of Neuroscience*, **18**(15): 5850-5858.

Elliott, T. and Shadbolt, N. (1999) 'A neurotrophic model of the development of the retinogeniculocortical pathway induced by spontaneous retinal waves', *The Journal of Neuroscience*, **19**: 951-7970.

Elliott, T. and Shadbolt, N. (2001) 'Growth and repair: Instantiating a biologically-inspired model of neuronal development on the Khepera robot', *Robotics and Autonomous Systems*, **36**: 149-169.

Eskiizmirliler, S., Maier, M.A., Zollo, L., Manfredi, L., Teti, G. and Laschi, C. (2006) 'Reach and Grasp for an Anthropomorphic Robotic System based on Sensorimotor Learning' in: *The proceedings of the First IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob 2006)*.

Espinosa, J. S. and Stryker, M. P. (2012) 'Development and Plasticity of the Primary Visual Cortex', *Neuron,* **75**: 230-249.

FACETS website (last updated Feb 2012) http://facets.kip.uni-heidelberg.de/

Fritzke, B. (1995) 'A growing neural gas network learns topologies' in: G. Tesauro, D. Touretzky, and T. Leen (eds.), *Advances in neural information processing systems*, **7**,

Cambridge, MA: MIT Press: 625–632.

Froemke, R. and Dan, Y. (2002) 'Spike-timing-dependent synaptic modification induced by natural spike trains', *Nature,* **416**: 433-438.

Galluppi, F., Brohan, K., Davidson, S., Serrano-Gottarredona, T., Corasco, J. P., Linares-Barranco, B. and Furber, S. (2012) 'A Real-Time, Event Driven Neuromorphic System for Goal-Directed Attentional Selection' in: *The Proceedings of the 19th International Conference on Neural Information processing (ICONIP 2012)*

Gamez, D. (2008) 'The Development and Analysis of Conscious Machines', PhD thesis, Department of Computing and Electronic Systems, University of Essex.

Gamez, D., Newcombe, R., Holland, O. and Knight, R. (2006) 'Two Simulation Tools for Biologically Inspired Virtual Robotics' in: *The Proceedings of the IEEE 5th Chapter Conference on Advances in Cybernetic Systems*.

Georgopoulos, A., Schwartz, A. and Kettner, R. (1986) 'Neuronal population coding of movement direction', *Science*, **233**(4771): 1416-1419.

Gerstner, W. (1999) 'Spiking Neurons' in: *Pulsed Neural Networks*, MIT Press: 1-53.

Gilbert, C. D. and Li, W. (2012) 'Adult Visual Cortical Plasticity', *Neuron*, **75**: 250-264.

Gómez, G., Lungarella, M., Eggenberger Hotz, P., Matsushita, K. and Pfeifer, R. (2004) 'Simulating development in a real robot: on the concurrent increase of sensory, motor, and neural complexity', in: L. Berthouze; H. Kozima; C. G. Prince; G. Sandini; G. Stojanov; Metta, G. and C. Balkenius, (eds), *Proceedings of the Fourth International Workshop on Epigenetic Robotics*.

Goodhill, G. (1993) 'Topography and ocular dominance: a model exploring positive correlations', *Biological Cybernetics,* **69:** 109-118.

Goodman, D.F. and Brette, R. (2008) 'Brian: a simulator for spiking neural networks in Python', *Frontiers in Neuroinformatics*, **2**.

Grossberg, S. (1975) 'On the Development of Feature Detectors in the Visual Cortex with Applications to Learning and Reaction-Diffusion Systems', *Biological Cybernetics,* **21**: 145-159.

Hagras, H., Pounds-Cornish, A., Colley, M., Callaghan, V. and Clarke, G. (2004) 'Evolving spiking neural network controllers for autonomous robots' in: *The proceedings of the IEEE International Conference on Robotics and Automation (ICRA04)*.

Harris, A., Ermentrout, G. and Small, S. (1997) 'A model of ocular dominance column development by competition for trophic factor', *PNAS,* **94**: 9944-9949.

Held, R., and Hein, A. (1967) 'Dissociation of the Visual Placing Response into Elicited and Guided Components', Science, **158**: 390-392.

Hines, M.L., Morse, T., Migliore, M., Carnevale, N.T. and Shepherd, GM. (2004) 'ModelDB: A Database to Support Computational Neuroscience', *The Journal of Computational Neuroscience*, **17**(1): 7-11.

Honda, M., Urakubo, H., Tanaka, K. and Kuroda, S. (2011) 'Analysis of Development of Direction Selectivity in Retinotectum by a Neural Circuit Model with Spike Timing-Dependent Plasticity', *Journal of Neuroscience,* **31**(4): 1516-1527.

Hubel, D.H. and Wiesel, T.N. (1977) 'Functional architecture of the macaque monkey visual cortex', Proc. R. Soc. Lond. B, **198**: 1-59

Hunt, J., Ibbotsen, M. and Goodhill, G. (2012) 'Sparse Coding on the Spot: Spontaneous Retinal Waves Suffice for Orientation Selectivity', *Neural Computation,* **24**(9): 2422-2433.

Ijspeert, A.J, Crespi, A., Ryczko, D. and Cabelguen, J-M. (2007) 'From swimming to walking with a salamander robot driven by a spinal cord model', Science, **315**: 1416–1419

iniLabs website (2010), 'DVS 128 Dynamic Vision Sensor', http://www.inilabs.com/products/

Izhikevich, E.M. (2003) 'Simple model of spiking neurons', *IEEE Transactions on Neural Networks*, **14**: 1569–1572

Izhikevich, E. M. (2004) 'Which Model to Use for Cortical Spiking Neurons?', *IEEE Transactions on Neural Networks*, **15**: 1063-1070.

Jaeger, H. (2002) 'Tutorial on Training Recurrent Neural Networks, Covering BPPT, RTRL, EKF and the 'Echo State Network' Approach': GMD-Forschungszentrum Informationstechnik.

jAER SourgeForge wiki (2012), http://sourceforge.net/apps/trac/jaer/wiki

Jin, X., Lujan, M., Plana, L., Davies, S., Temple, S. and Furber, S. (2010) 'Modeling Spiking Neural Networks on SpiNNaker', *Computing in Science & Engineering*, **21**(5): 91-97.

Jun, J. and Jin, D. (2007) 'Development of Neural Circuitry for Precise Temporal Sequences through Spontaneous Activity, Axon Remodeling, and Synaptic Plasticity', *PLoS One*, **8**(2), e723. doi:10.1371/journal.pone.0000723.

Kandel, E.R., Schwartz, J.H. and Jessell, T.M. (2000), *Principles of Neural Science* (Fourth ed.). McGraw Hill.

Kaschube, M., Schnabel, M., Lowel, S., Coppola, D., White, L. and Wolf, W. (2010) 'Universality in the Evolution of Orientation Columns in the Visual Cortex', *Science,* **330**(6007): 1113-1116.

Keith-Magee, R.; Venkatesh, S.; Takatsuka, M. (1999) 'An empirical study of neighbourhood decay in Kohonen's self-organising map' in: *The Proceedings of the International Joint Conference on Neural Networks (IJCNN '99):* 1953-1958

Kikuchi, M., Ogino, M. and Asada, M. (2004) 'Visuo-motor Learning for Behavior Generation of Humanoids' in: *The Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*: 521-526.

Kim, D. (2006) 'Neural network mechanism for the orientation behavior of sand scorpions towards prey', *IEEE Transactions on Neural Networks*, **17**: 1070–1076.

Kohonen, T. (1984), *Self-organisation and Associative Memory,* Springer-Verlag, Berlin

Kohonen, T. (1995), *Self-organizing Maps*, Springer-Verlag, Berlin.

Krichmar, J. and Edelman, G. (2003) ‘Brain-Based Devices: Intelligent Systems Based on Principles of the Nervous System’ in: *The Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Krose, B. and Eecen, M. (1994) ‘A Self-Organizing Representation of Sensor Space for Mobile Robot Navigation’ in: *The Proceedings of the 1994 IEEE IEEE/RSJ Int. Conf. on intelligent Robots and Systems*: 9-14.

Kuperstein, M. (1988) ‘Neural model of adaptive hand-eye coordination for single postures’, *Science, **239***: 1308–1311.

Land, M. (1972) ‘Stepping movements made by jumping spiders during turns mediated by the lateral eyes’, *Journal of Experimental Biology*, **57**: 15–40.

Lapicque, L. (1907) ‘Recherches quantitatives sur l'excitation electrique des nerfs traitée comme une polarisation’, *J. Physiol. Pathol. Gen*., **9**: 620-635.

Linares-Barranco, A., Gomez-Rodriguez, F., Jimenez-Fernandez, A., Delbrück, T. and Lichtensteiner, P. (2007) ‘Using FPGA for visuo-motor control with a silicon retina and a humanoid robot’, in: *The Proceedings of the IEEE Symposium on Circuits and Systems (ISCAS 2007)*: 1192 - 1195.

Maass, W. (1997) 'Networks of spiking neurons: The third generation of neural network models', *Neural Networks,* **10**: 1659–1671.

Marian, I. (2002) 'A biologically inspired model of motor control of direction', Master's thesis, University College Dublin.

McCulloch, W. and Pitts, W. (1943) ‘A logical calculus of the ideas immanent in nervous activity’, *Bulletin of Mathematical Biophysics*, **7**: 115 - 133.

Metta, G., Sandini, G. and Konczak, J. (1999) 'A developmental approach to visually-guided reaching in artificial systems', *Neural Networks,* **12**(10): 1413-1427.

Meyer, J., Guillot, A., Girard, B., Khamassi, M., Pirim, P. and Berthoz, A. (2005) 'The Psikharpax project: Towards building an artificial rat', *Robotics and Autonomous Systems,* **50**(4): 211-223.

Miikkulainen, R., Bednar, J., Choe, Y. and Sirosh, J. (1998) 'A self-organizing neural network model of the primary visual cortex' in: *The Proceedings of the Fifth International Conference on Neural Information Processing*: 815-818.

Miikkulainen, R.; Bednar, J.; Choe, Y. and Sirosh, J. (2005), *Computational Maps in the Visual Cortex*, Springer-Verlag, New York.

Miyoshi, T. (2005) ‘Initial Node Exchange and Convergence of SOM Learning’ in: *The Proceedings of the 6th Symposium on Advanced Intelligent Systems.*

Miyoshi, T. (2007) 'Neighbor Size of Initial Node Exchange and its Influence for SOM Learning', *Journal of Advanced Computational Intelligence and Intelligent Informatics,* **11**(6): 620-625.

Miyoshi, T. (2008) 'Relation Organization of SOM Initial Map by Improved Node Exchange', *Journal of Computers,* **3**(9): 77-84.

Morse, A. and Ziemke, T. (2009) ‘Action, Detection, and Perception: A Computational Model of the Relation Between Movement and Orientation Selectivity in the Cerebral Cortex’ in: *The Proceedings of the 31th Annual Conference of the Cognitive Science Society*.

Mulier, F. and Cherkassy, V. (1994) 'Learning rate schedules for self-organizing maps' in: The Proceedings of the 12th IAPR International Conference on Pattern Recognition, Vol. 2 - Conference B: Computer Vision and Image Processing.

Mussa-Ivaldi, F. A. (1992) 'From basis functions to basis fields: vector field approximation from sparse data', *Biological Cybernetics,* **67:** 479-489.

Mussa-Ivaldi, F. A. and Giszter, S. (1992) 'Vector field approximation: a computational paradigm for motor control and learning', *Biological Cybernetics,* **67:** 491-500.

Mussa-Ivaldi, F. A., Giszter, S. and Bizzi, E. (1994) 'Linear combinations of primitives in vertebrate motor control', *PNAS,* **91:** 7534-7538.

Ogino, M., Kikuchi, M., Ooga, J., Aono, M. and Asada, M. (2005) 'Optic flow based skill learning for a humanoid to trap, approach to, and pass a ball', *RoboCup 2004: Robot Soccer World Cup VIII, volume 3276 of LNCS*, Springer-Verlag: 323–334.

Paine, R. and Tani, J. (2004) 'Motor primitive and sequence self-organization in a hierarchical recurrent neural network', *Neural Networks*, **17**: 1291–1309.

Panchev, C. and Wermter, S. (2001) 'Hebbian spike-timing dependent self-organization pulse neural networks' in: *Proceedings of World Congress on Neuroinformatics*: 378-385.

Pham, D., Packianather, M. and Charles, E. (2006) 'A Novel Self-Organised Learning Model with Temporal Coding for Spiking Neural Networks' in: *Proceedings of Intelligent Production Machines and Systems*.

Pulvermüller, F. (2005) 'Brain mechanisms linking language and action', *Nature Reviews Neuroscience*, **6**(7): 576-582.

Raginsky, M. and Anastasio, T. (2008) 'Cooperation in self-organizing map networks enhances information transmission in the presence of input background activity', *Biological Cybernetics,* **98:** 195-211.

Ritter, H., Martinez, T. and Schulten, K. (1989) 'Topology-conserving maps for learning visuo-motor coordination', *Neural Networks,* **2:** 159-168.

Robotis Website (undated). Available online: http://www.robotis.com/xe/bioloid_en

Rochefort, N., Narushima, M., Grienberger, C., Marandi, M., Hill, D. and Konnerth, A. (2011) 'Development of Direction Selectivity in Mouse Cortical Neurons', *Neuron,* **71**: 425-432.

Root, T.M. (1990) 'Chapter 9: Neurobiology', in: G.A. Polis (Ed.), *The Biology of Scorpions*, Stanford University Press, Stanford, California: 341–411.

Ruf, B. and Schmitt, M. (1998) 'Self-organization of spiking neurons using action potential timing', *IEEE Transactions on Neural Networks,* **9**: 575-578.

Sala, D., Cios, K. and Wall, J. (1998) 'Self-organization in networks of spiking neurons', *Australian Journal of Intelligent Information Processing Systems,* **5**(3): 161-170.

Schmidt, J. (1985) 'Formation of retinotopic connections: Selective stabilization by an activity dependent mechanism', *Cellular and Molecular Neurobiology,* **15**(1-2): 65-84.

SDL Component Suite Website (Last Updated 2008), "Kohonen Network - Background Information", http://www.lohninger.com/helpcsuite/kohonen_network__background_information.htm. Accessed online July 2012.

Serrano-Gotarredona, R., Oster, M., Lichtsteiner, P., Linares-Barranco, A., Paz-Vicente, R., Gomez-Rodriguez, F., Camunas-Mesa, L., Berner, R., Rivas, M., Delbrück, T., Liu, S-C., Douglas, R., Hafliger, P., Jimenez-Moreno, G., Civit, A., Serrano-Gotarredona, T., Acosta-Jimenez, A. and Linares-Barranco, B. (2009) 'CAVIAR: A 45k-neuron, 5M-synapse, 12G-connects/sec AER hardware sensory-processing-learning-actuating system for high speed visual object recognition and tracking', *IEEE Transactions on Neural Networks,* **20**: 1417-1438.

Shah-Hosseini, H. and Safabakhsh, R. (2000) 'TASOM: the time adaptive self-organizing map' in: *The Proceedings of the IEEE International Conference on Information Technology: Coding and Computing*, Las Vegas, Nevada: 422–427.

Shah-Hosseini, H. and Safabakhsh, R. (2001) 'Automatic Adjustment of Learning Rates of the Self-Organising Feature Map', *Scientia Iranica*, **8**(4): 277-286.

Shah-Hosseini, H. (2011) 'Binary tree time adaptive self-organizing map', *Neurocomputing*, **74**: 1823-1839.

Shon, A., Rao, R. and Sejnowski, T. (2004) 'Motion detection and prediction through spike-timing dependent plasticity', *Network: Computation in Neural Systems,* **15**: 179-198.

Shultz, J. (1987) 'Walking and surface film locomotion in terrestrial and semi-aquatic spiders', *Journal of Experimental Biology,* **128**: 427–444.

Silver, R., Boahen, K., Grillner, S., Kopell, N. and Olsen, K.L. (2007) 'Neurotech for neuroscience: Unifying concepts, organizing principles, and emerging tools', *Journal of Neuroscience*, **27**: 11807-11819.

Song, S., Miller, K. D. and Abbott, L. F. (2000) 'Competitive Hebbian learning through spike-timing dependent synaptic plasticity', *Nature Neuroscience*, **3**: 919-926.

Song, S. and Abbott, L. F. (2001) 'Cortical Development and Remapping through Spike Timing-Dependent Plasticity', *Neuron,* **32**: 339-350.

Stratton, P., Milford, M., Wiles, J. and Wyeth, G. (2009) 'Automatic Calibration of a Spiking Head-Direction Network for Representing Robot Orientation' in: *The Proceedings of the 2009 Australasian Conference on Robotics and Automation.*

Stürzl, W., Kempter, R. and van Hemmen, J. (2000) 'Theory of Arachnid Prey Localization', *Physical Review Letters,* **84**(24): 5668-5671.

Terada, K., Takeda, H. and Nishida, T. (1998) 'An Acquisition of the Relation between Vision and Action using Self-Organizing Map and Reinforcement Learning' in: *The Proceedings of the Second International Conference on Knowledge-Based Intelligent Electronic Systems, (KES'98)*.

Thorpe, S. J., Fize, D. and Marlot, C. (1996) 'Speed of processing in the human visual system', *Nature*, **381**: 520-522.

Thorpe, S. J., Delorme, A. and VanRullen, R. (2001) 'Spike-based strategies for rapid processing', *Neural Networks,* **14**(6-7): 715-726.

Toussaint, M. (2006) 'A sensorimotor map: Modulating lateral interactions for anticipation and planning', *Neural Computation,* **18**: 1132-1155.

Toussaint, M. (2004) 'Learning a World Model and Planning With a Self-Organizing Dynamic Neural System' in: *The Proceedings of Advances in Neural Information Processing Systems 16*: 926-936.

van Ooyen, A. (2001) 'Competition in the development of nerve connections: a review of models', *Network: Computation in Neural Systems,* **12**: 1-47.

van Rossum, M. (2001) 'A Novel Spike Distance', *Neural Computation,* **13**: 751–763.

van Rossum, M., Bi, G. and Turrigiano, G. (2000) 'Stable Hebbian Learning from Spike Timing-Dependent Plasticity', *The Journal of Neuroscience,* **20**(23): 8812-8821.

Vogels, T.P. and Abbott, L.F. (2005) 'Signal Propagation and Logic Gating in Networks of Integrate-and-Fire Neurons', *The Journal of Neuroscience,* **25**(46): 10786-10795

Wang, X., Zeng-Gouang, H., Zou, A., Tan, M. and Cheng, L. (2008) 'A behavior controller based on spiking neural networks for mobile robots', *Neurocomputing,* **71**: 655-666.

Weliky, M., Bosking, W.H. and Fitzpatrick D. (1996) 'A systematic map of direction preference in primary visual cortex', *Nature,* **379**: 725–728.

Wenisch, O., Noll, J. and van Hemmen, J. L. (2005) 'Spontaneously emerging direction selectivity maps in visual cortex through STDP', *Biological Cybernetics,* **93**: 239-247.

White, B., Castle, P. and Held, R. (1964) 'Observations on the development of visually guided reaching', *Child Development*, ***35***: 349–364.

Willshaw, D.J. and von der Malsburg, C. (1976) 'How patterned neural connections can be set up by self-organisation', *Proc. R. Soc. Lond. B.*, **194**: 431–445.

Zollo, L., Guglielmelli, S., Teti, G., Laschi, C., Eskiizmirliler, S., Carenzi, F., Bendahan, P., Gorce, P., Maier, M.A., Burnod, Y. and Dario, P. (2005) 'A Bio-inspired Neuro-Controller for a Anthropomorphic Head-Arm Robotic System' in: *The Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA 2005).*

Zollo, L., Eskiizmirliler, S., Teti, G., Laschi, C., Burnod, Y., Guglielmelli, E. and Maier, M.A. (2008) 'An Anthropomorphic Robotic Platform for Progressive and Adaptive Sensorimotor Learning', *Advanced Robotics,* **22**(1): 91-118.